104

SOLVING THE YEAR 2000 SOFTWARE PROBLEM: CREATING BLUEPRINTS FOR SUCCESS

Y 4.SCI 2:104/48

Solving the Year 2000 Softuare Prob...

HEARING

BEFORE THE

SUBCOMMITTEE ON TECHNOLOGY

OF THE

COMMITTEE ON SCIENCE

U.S. HOUSE OF REPRESENTATIVES

ONE HUNDRED FOURTH CONGRESS

SECOND SESSION

MAY 14, 1996

[No. 48]

Printed for the use of the Committee on Science

DEPOSITORY

SEP 1 8 1996

U.S. GOVERNMENT PRINTING OFFICE
26–019CC WASHINGTON : 1996

For sale by the U.S. Government Printing Office
Superintendent of Documents, Congressional Sales Office, Washington, DC 20402
ISBN 0-16-052913-1

104

SOLVING THE YEAR 2000 SOFTWARE PROBLEM: CREATING BLUEPRINTS FOR SUCCESS

Y 4.SCI 2:104/48

Solving the Year 2000 Softuare Prob...

HEARING

BEFORE THE

SUBCOMMITTEE ON TECHNOLOGY

OF THE

COMMITTEE ON SCIENCE

U.S. HOUSE OF REPRESENTATIVES

ONE HUNDRED FOURTH CONGRESS

SECOND SESSION

MAY 14, 1996

[No. 48]

Printed for the use of the Committee on Science

DEPOSITORY

SEP 18 1996

U.S. GOVERNMENT PRINTING OFFICE
26–019CC WASHINGTON : 1996

For sale by the U.S. Government Printing Office
Superintendent of Documents, Congressional Sales Office, Washington, DC 20402
ISBN 0-16-052913-1

COMMITTEE ON SCIENCE

ROBERT S. WALKER, Pennsylvania, *Chairman*

F. JAMES SENSENBRENNER, Jr., Wisconsin
SHERWOOD L. BOEHLERT, New York
HARRIS W. FAWELL, Illinois
CONSTANCE A. MORELLA, Maryland
CURT WELDON, Pennsylvania
DANA ROHRABACHER, California
STEVEN H. SCHIFF, New Mexico
JOE BARTON, Texas
KEN CALVERT, California
BILL BAKER, California
ROSCOE G. BARTLETT, Maryland
VERNON J. EHLERS, Michigan**
ZACH WAMP, Tennessee
DAVE WELDON, Florida
LINDSEY O. GRAHAM, South Carolina
MATT SALMON, Arizona
THOMAS M. DAVIS, Virginia
STEVE STOCKMAN, Texas
GIL GUTKNECHT, Minnesota
ANDREA H. SEASTRAND, California
TODD TIAHRT, Kansas
STEVE LARGENT, Oklahoma
VAN HILLEARY, Tennessee
BARBARA CUBIN, Wyoming
MARK ADAM FOLEY, Florida
SUE MYRICK, North Carolina

GEORGE E. BROWN, Jr., California RMM*
HAROLD L. VOLKMER, Missouri
RALPH M. HALL, Texas
BART GORDON, Tennessee
JAMES A. TRAFICANT, Jr., Ohio
JOHN S. TANNER, Tennessee
TIM ROEMER, Indiana
ROBERT E. (Bud) CRAMER, Jr., Alabama
JAMES A. BARCIA, Michigan
PAUL McHALE, Pennsylvania
JANE HARMAN, California
EDDIE BERNICE JOHNSON, Texas
DAVID MINGE, Minnesota
JOHN W. OLVER, Massachusetts
ALCEE L. HASTINGS, Florida
LYNN N. RIVERS, Michigan
KAREN McCARTHY, Missouri
MIKE WARD, Kentucky
ZOE LOFGREN, California
LLOYD DOGGETT, Texas
MICHAEL F. DOYLE, Pennsylvania
SHEILA JACKSON LEE, Texas
WILLIAM P. LUTHER, Minnesota

DAVID D. CLEMENT, *Chief of Staff and Chief Counsel*
BARRY BERINGER, *General Counsel*
TISH SCHWARTZ, *Chief Clerk and Administrator*
ROBERT E. PALMER, *Democratic Staff Director*

SUBCOMMITTEE ON TECHNOLOGY

CONSTANCE A. MORELLA, Maryland, *Chairman*

SUE MYRICK, North Carolina
KEN CALVERT, California
GIL GUTKNECHT, Minnesota
ANDREA H. SEASTRAND, California
TODD TIAHRT, Kansas
BARBARA CUBIN, Wyoming

JOHN S. TANNER, Tennessee
PAUL McHALE, Pennsylvania
EDDIE BERNICE JOHNSON, Texas
KAREN McCARTHY, Missouri
ZOE LOFGREN, California

---

*Ranking Minority Member
**Vice Chairman

CONTENTS

WITNESSES

Page

May 14, 1996:
- Peter De Jager, President/Owner, de Jager & Company, Brampton, Ontario, Canada 3
- D. Dean Mesterharm, Deputy Commissioner for Systems, Social Security Administration, Baltimore, Maryland 6
- Robert Hebner, Acting Deputy Director, National Institute of Standards and Technology, Gaithersburg, Maryland 18
- Barry Ingram, Chief Technical Officer, EDS Government Services Group, Herndon, Virginia 22
- Barbara McDuffie, Program Director, System 390, Solution Provider Program, IBM, Poughkeepsie, New York 30
- Marc Sokol, Vice President of Advanced Technologies, Computer Associates, Islandia, New York 253

APPENDIX

Statement of William Ulrich, President and Founder of Tactical Strategy Group, Inc 278

Statement of Terry J. Piddington, Executive Vice President, CTA INCORPORATED, Rockville, Maryland 279

SOLVING THE YEAR 2000 SOFTWARE PROBLEM: CREATING BLUEPRINTS FOR SUCCESS

TUESDAY, MAY 14, 1996

U.S. HOUSE OF REPRESENTATIVES,
COMMITTEE ON SCIENCE,
SUBCOMMITTEE ON TECHNOLOGY,
Washington, DC.

The Subcommittee met at 1:09 p.m. in Room 2325 of the Rayburn House Office Building, the Honorable Constance A. Morella, Chairwoman of the Subcommittee, presiding.

Chairwoman MORELLA. The hearing will come to order.

I apologize for being a little late. I was meeting with 150 constituents from Montgomery County, Maryland.

John Tanner says that's the advantage of coming from Tennessee. They can't get here quite as often. So, I thank you for waiting.

I am pleased to convene this hearing on the Year 2000 software problem. Today, we will review the origins and the extent of the Year 2000 problem and discuss strategies and tools that can be employed to facilitate a solution.

The Year 2000 is rapidly approaching. The next millennium is expected to be a time of great change. Unfortunately, a vast majority of our nation's computer systems are not equipped to handle the simple change of date initiated by the turn of the century.

Most of the computer software in use today employs two digit date fields. Consequently, at the turn of the century, computer software will be unable to differentiate between the years 1900 and 2000.

If this software system is not addressed promptly, it will render the vast majority of date sensitive computer information unusable and obsolete. The Year 2000 problem affects computer programs in virtually every government agency. It has the potential to adversely affect government transfer payments, tax collection and critical military missions.

And, in addition, the Year 2000 problem will affect virtually every private enterprise. Even companies that are using Year 2000 compliance software, or no computer system at all, are still likely to be affected by suppliers and customers whose systems must be changed.

Solutions to this problem are both managerial and technical. And, today we will hear from experts in government and the private sector who are working to find the most efficient means of addressing the Year 2000 software problem.

I look forward to their testimony. And, maybe instead of calling it "computer problem," we should call it "computer challenge."

I would now like to yield to the Ranking Member of the Subcommittee, Mr. Tanner.

Mr. TANNER. Thank you, Ms. Morella. And, I want to thank you for holding this.

This is not unlike many of the issues that we have discussed in the Science Committee from time to time, in that although it is sometimes not very well publicized, a lot of the problems we discuss affect literally the everyday life of every citizen in this country. And, this certainly would be, or could be, one of those.

I didn't know this was going to be such a monumental task some short time ago. And, probably in that respect, I wasn't unlike many of our friends and neighbors across this country who maybe haven't given much thought to the problem.

I think I have learned that it is a fairly simple one to understand and really, in principle, not terribly difficult or impossible to correct. But, what we do find is that we not only have an opportunity to educate people, both in the public and more particular in the private sector where they have maybe not the level of awareness that some of us who are in the public sector and have time to study these issues possess, but we understand that in fixing the problem there is going to have to be some attention given to how to do it, given the fact that there are so many different programs out there.

That complicates hugely, and I would assume would make it more expensive, whatever has to be done, particularly in the private sector. And, so I want to join Mrs. Morella and thank all of you who are here.

This is a nice sized crowd. And, I want to thank our witnesses certainly for your time and attention to this important matter.

And, I look forward to hearing from you.

Chairwoman MORELLA. Thank you. We have been joined by the Vice Chairman of the Subcommittee, Mr. Calvert.

Mr. CALVERT. Thank you, Madam Chair. I would like to take the opportunity to thank you once again for holding a hearing on what is turning out to be a very important subject.

Prior to today, I'm sure many people in this country are unaware that there will be a software problem in the various government and private sector computer systems by the Year 2000. Millions of people could conceivably be affected by this problem, maybe even affecting someone's risk of one's health.

Without a doubt, if this problem is not addressed and solved in a timely manner, the vast majority of date sensitive computer information will become obsolete and unusable. Patients who were born in the early 1920s and who regularly receive medical reminders for checkups as a result of being over the age of 65 will, instead, be recognized by doctors' computer software as being only in their mid-20s.

Chairwoman MORELLA. That would apply to me.

[Laughter.]

Mr. CALVERT. Yeah, I was going to say, let's just leave it the way it is. Probably, though, this is not acceptable.

So, I appreciate the hurdles ahead of us in trying to solve this problem. It's my understanding from a memo that I read that the estimates to fix this could run about $600 billion. And, so since we don't have the $600 billion, as you are hearing around here a lot, we are going to have to find a way to fix this with less money.

I hope the panelists can shed more light on this subject and hope the Subcommittee, led by you, Chairwoman Morella, can contribute in some way in finding a solution. I want to thank you again and welcome Mr. de Jager, Mr. Mesterharm of Panel One; and, Dr. Hebner, Mr. Ingram, Ms. McDuffie and Mr. Sokol of Panel Two. And, I will stay as long as I can.

Chairwoman MORELLA. Thank you, Mr. Calvert. I know of your interest in it.

I would like to now proceed with the first panel. And, I want to again thank Mr. Peter de Jager, who is the President/Owner of de Jager and Company, Brampton, Ontario. Thank you for being here.

Mr. Dean Mesterharm, who is the Deputy Commissioner for Systems, Social Security Administration in Baltimore, Maryland, the home of the Baltimore Ravens, who is also accompanied by Kathleen Adams, also of the Social Security Administration. Thank you.

We do have written copies of your testimony. If you would like to abbreviate, we will try to look at the five minute rule if possible.

Thank you.

STATEMENT OF PETER de JAGER, PRESIDENT/OWNER, de JAGER AND COMPANY, CANADIAN COMPUTER MANAGEMENT CONSULTANT, BRAMPTON, ONTARIO

Mr. DE JAGER. Computer practitioners are the most optimistic people in the world. Despite all evidence to the contrary, we believe the next application we write will be bug free.

We believe the bug we just found will be the last one. And, we believe the next release of the software product will solve all errors in the prior release and introduce no new ones.

Sadly, these beliefs are totally without foundation. We have a reputation of never meeting deadlines.

Have we earned this reputation? The facts speak for themselves.

According to a study done by Capers Jones, fewer than 14 percent of projects larger than 100,000 function points in size—that's about 12.5 million lines of C code—are delivered on time. Fewer than 14 percent.

We also happen to believe we can solve the Year 2000 problem in time. This project is unique.

First, the deadline cannot be missed. Second, it's an immovable deadline.

When the Year 2000 arrives, programs we used yesterday will be useless. Unless the applications are fixed and available on January the 1st, Year 2000, all businesses lose the ability to do business.

I am at a loss as to how to communicate that message any simpler.

Thirdly, this task, this deadline, bears no relationship to the size of the task. Regardless of how many programs you have to fix, the deadline is the same.

Usually, we set deadlines by the size of the task. The nature of this problem is such that it removes the necessity for that part of the planning process.

The deadline we all share is January the 1st, the Year 2000.

You will have heard from some witnesses that you can rest assured they will complete the project on time. But, this is nothing more than unjustified optimism.

You must weigh their testimony against the industry track record of delivering projects on time. You must then adjust their testimony further to take into account the following realities.

They cannot tell you in detail how large their task is. They cannot tell you when their software vendors will be Year 2000 compliant, because the vast majority of the vendors have not yet disclosed these delivery dates.

They cannot tell you when their business partners will be changing their data formats and how those data formats will change.

And, finally and perhaps most importantly, we come to the fourth unique aspect of this problem. We all share exactly the same deadline. What this means is that we are all going after the same experts.

We are going after the same project leaders, the same vendors. We will be raiding each other for expertise.

The absolute worse thing that can happen to a project team is to lose key team members. Any project subject to this risk has no right to claim we can rest assured that they will deliver on time.

The situation is critical. More than 65 percent of North American businesses have not yet begun to address this problem. There are less than 140 weekends left before December the 31st, 1998.

We must be complete with our development maintenance changes by then so that we have all of 1999 to test the hundreds of thousands of changes we've put in. Sixty-five percent of North American businesses are totally unaware of even this minimal planning strategy.

We have no time for unjustified optimism nor have we time for cautious optimism. We have only time for a highly accelerated sense of urgency and a meager allotment of time rapidly slipping away.

If we have any hope of delivering on time in the future, despite our record of delivering late in the past, then we must replace unjustified optimism with determined urgency. And, along with that, I wish us luck. We are going to need it.

[The prepared statement of Mr. de Jager follows:]

STATEMENT ON:

THE YEAR 2000 COMPUTER CRISIS

BY

PETER DE JAGER, CANADIAN COMPUTER MANAGEMENT CONSULTANT

BEFORE THE COMMITTEE ON GOVERNMENT REFORM AND OVERSIGHT SUBCOMMITTEE ON GOVERNMENT MANAGEMENT, INFORMATION AND TECHNOLOGY HOUSE OF REPRESENTATIVES

MAY 14th 1996

Unjustified Optimism

Computer Practitioners are the most optimistic people in the world. Despite all evidence to the contrary we believe the next application we write, will be bug free. We believe the bug we just found, is the last one. We believe the next release of a software product will solve all the errors in the prior release and introduce no new ones.

Sadly, these beliefs are totally without foundation. Our clients know this. We have a reputation of always missing delivery deadlines. Have we earned this reputation? The facts speak for themselves. According to studies done by Capers Jones, fewer than 14% of projects larger then 100,000 function points in size (about 12,500,000 fines of C code) are delivered on time.[1]

We also believe we can solve the Year 2000 problem in time.

The Year 2000 project is unique, four elements of that uniqueness are;

1) The deadline cannot be missed.

2) It is an immovable deadline.

In the past, if we missed a delivery date, we could continue to use what we used yesterday. When the Year 2000 arrives, the programs we used yesterday will be useless. Unless the applications are fixed and available on January 1st, all businesses lose the ability to do business. I am at a loss as to how to communicate that message any simpler.

I will leave it to you to contemplate what happens to the world-wide economy if businesses lose the ability to do business.

3) It bears no relationship to the size of the task.

Regardless if you have a single program to fix, or 75,000 programs to fix, the deadline is the same. Usually we set deadlines by the size of the task and how long we estimate it will take to complete that task with available resources. The nature of this problem removes that part of the planning process. The deadline is January 1st. 2000.

You will have heard from some witnesses that you can 'rest assured' they will complete this project on time. This is nothing more than unjustified optimism. You must weigh their testimony against their past track record of delivering on time. You must then adjust their testimony further, to take into account the following realities.

a) They cannot tell you in detail how large their task is.
e.g.. How many lines of code or applications they manage.

b) They cannot tell you when their software vendors will be year 2000 compliant, because the vast majority of vendors have not yet disclosed these release schedules.
e.g.. Operating systems, system utilities, 3rd party applications

c) They cannot tell you when their business partners will be changing their data formats and how will those data formats change.
e.g.. When will the Federal Reserve Bank change from 2 to 4 digit years?

And finally, and perhaps most importantly, we come to the 4th unique aspect of this problem.

4) We share the same deadline.

This adds a very large, unpredictable and non-technical complication to the problem. What will organizations do, to make sure they don't miss a deadline they cannot afford to miss? They'll want to hire the best and will be willing to pay whatever is required to get them. Let's rephrase that. They'll raid other organizations for the best, most skilled, most respected.

The worst thing to happen to a project team working to a deadline that cannot be missed is to lose the key team members. Any project subject to this risk has no right to claim we can 'rest assured'.

The situation is critical. More than 65% of North American businesses have not yet begun to address this problem. For many it's already too late. There are less than 140 weekends left before December 31st 1998. You should be complete by then, so that you can allocate all of 1999 to test the hundreds of thousands of error prone changes you've introduced into your systems.

65% of North American businesses are unaware of even this minimal planning strategy.

We have no time for 'Unjustified Optimism.' Nor have we time for cautious optimism. We have time only for a highly accelerated sense of urgency, a meager allotment of time rapidly slipping away.

I have kept my testimony brief, because I want to keep my message as concise and as clear as possible.

a) The deadline is real, immovable and cannot be missed.
b) We have less than 140 weekends left to complete the task.
c) Less than 35% of North American Businesses have begun.
d) Those active find this to be the most complex project they've ever attempted.

---

[1] Large Software System Failures and Successes by Capers Jones, American Programmer, April 1996, Vol 9 no 5

e) There are unique non-technical obstacles in our path.
f) The IS community suffers under a delusion of infallible confidence, despite a proven track record of on-time delivery no greater than 14%
g) The sense of urgency required to complete this task on time is absent.

I wish to thank you for the opportunity to testify on this matter, I hope that my testimony today, contributes in some small way to the arguments and testimony already presented.

If we have any hope of delivering on time in the future, despite our record of delivering late in the past, then we must replace unjustifiable optimism with determined urgency.

Along with that, I wish us luck, we're going to need it.

Chairwoman MORELLA. Pretty succinct and pretty definite. Thank you, Mr. de Jager.

Mr. Mesterharm.

STATEMENT OF D. DEAN MESTERHARM, DEPUTY COMMISSIONER FOR SYSTEMS, SOCIAL SECURITY ADMINISTRATION, BALTIMORE, MARYLAND

Mr. MESTERHARM. Madam Chairwoman and members of the Subcommittee, I appreciate the opportunity to discuss the Year 2000 project at the Social Security Administration. Madam Chairwoman, a unique event will take place on January 1st, 2000.

On that day, we will experience the first century change since the start of the computer era. This event poses enormous challenges for the data processing community, as public and private sector organizations around the world prepare for the single largest integration feat since computers entered our daily lives.

The solution to the problem is obvious but labor intensive. Anywhere in our computer programs where we are currently adding, subtracting, comparing or sorting, using a two digit year, we need to substitute a four digit year.

While that sounds simple, our experience shows us that implementing it is far from a simple task. There is no way to create a technical quick fix to this problem. Each line of computer program must be examined individually to see if a change is needed.

Of course, we are not the only organization faced with this problem or the only one involved in addressing it. Every organization in the world, including every federal and state agency, that uses computers must address this very problem.

I am proud to report that SSA is in the forefront in planning for and dealing with this issue. We have changed the formats of our dates in our major data bases to include the century and have begun making changes to our application software.

We will have all 2000—all Year 2000 changes made by December 31st, 1998.

I would like to briefly mention that the Office of Management and Budget asked SSA last year to lead interagency discussions designed to increase awareness of the Year 2000 issue and encourage a sense of urgency. The Year 2000 Interagency Committee is chaired by Ms. Kathleen Adams.

The Committee began meeting in November 1995 with only a few representatives attending. But, the number has grown to 31 departments and agencies. And, the number grows with every meeting.

The Interagency Committee sponsored a Year 2000 conference for all government agencies on May 2nd, 1996 in Washington. Approximately 500 people from government agencies and private sector firms attended the conference and shared their experiences and approaches for addressing the Year 2000 problem in their organization.

In conclusion, Madam Chairwoman, it would be impossible to overstate the importance of a smooth and timely transition of computer operations to accommodate the need to reflect four digit years as we enter the 21st century.

[The prepared statement of Mr. Mesterharm follows:]

STATEMENT ON THE YEAR 2000

BY

MR. D. DEAN MESTERHARM, DEPUTY COMMISSIONER FOR SYSTEMS, SOCIAL SECURITY ADMINISTRATION

Madame Chairman and Members of the Subcommittee:

I appreciate the opportunity to discuss the Year 2000 Project at the Social Security Administration (SSA), and I thank you for your efforts to focus attention on a matter which urgently needs to be addressed. In your letter of invitation you asked me to describe what we are doing to prevent breakdowns in our systems and whether Federal agencies are aware of the problem. I will begin by outlining the reasons why change is needed and what we are doing to address the problem, so that the transition to the new century is a smooth one.

Reason Change Is Needed

Mr. Chairman, a unique event will take place on January 1, 2000. On that day, we will experience the first century change since the start of the computer era. This event poses enormous challenges for the data processing community, as public and private sector organizations around the world prepare for the single largest integration feat since computers entered our daily lives.

The reason that the century change poses a problem is that many computer programs store and use only the last two digits of a year and assume that the first two digits are 19. Under this practice, computer logic operations work as long as dates are in the same century, but problems arise when it is necessary to use dates in two different centuries. For example, subtracting December 31, 1995, from December 31, 2005, to determine someone's age would produce the incorrect answer of minus 90 instead of the correct result of 10.

The interaction of dates among different programs, systems, and agencies is one of the factors which gives the Year 2000 issue such complexity. Timing considerations become very important because either the sending or receiving agency will need to convert files from one format to another, unless both are ready to make their Year 2000 changes at the same time. For example, every employer in the United States with 250 or more employees must report their employees' earnings to SSA in some form of magnetic media. It is unlikely that they will all be converted at the same time. It is more likely that they will process their reports through a filtering program to substitute the appropriate date format.

Labor-Intensive Process

The solution to the problem is obvious, but labor intensive, for organizations such as SSA which depend heavily on computer operations. Wherever we currently add, subtract, compare, or sort using a two-digit year, we will need to substitute a four digit year.

While that sounds simple, our experience shows us that implementing it in computer systems is far from a simple task. There is no way to create a technical quick-fix to this problem. Virtually all computer systems rely on dates to some extent, but agencies such as SSA which are extremely date-sensitive are at greater risk if the Year 2000 conversion is not done on time and properly. At SSA, there are two complicating factors in the conversion process. One is the sheer size of the task. SSA has over 30 million lines of software now in use. The other is that there is no automated way to review the software. Each line must be examined individually to see if a change is needed. Our initial estimates indicate that it will take approximately 300 workyears to make and test the necessary changes, and the entire effort throughout SSA could require many more workyears. We are currently in the process of refining our estimates of total workyears which will be needed. Regardless of the amount of workyears needed, this activity cannot be deferred. We are planning to complete this project with in-house resources, but that means that, if additional resources are not budgeted, the resources for this critical project will not be available to do other systems development and modernization projects that would assist in processing increasing workloads with decreasing staff.

Of course, we are not the only organization faced with this problem, or the only one involved in addressing it. Every organization in the world, including every Federal and State agency that uses computers, must address this very problem. I am proud to report that SSA is in the forefront in planning for and dealing with this issue. In fact, SSA began examining the problem in 1989. We have changed the formats of dates in our major data bases to include the century and have begun making changes to our application software. All of the new software we are developing is, of course, year 2000-compliant.

As part of our early efforts, we conducted pilots, involving representative programs, and studied the time required to modify them for date changes. These pilots raised the awareness of our personnel of the amount of time and complexity the entire project would entail, and gave us an indication of how to schedule the work to be done.

We will have all Year 2000 changes made by December 31, 1998. This will give us an entire year to use our millennium changes in our production systems, ensuring that our current processing is unaffected and that the Year 2000 changes also function as designed. Of course, while we are making these changes to all our systems, our other work must remain on schedule.

After all, we must have the changes working by January 1, 2000. Unlike other computer outages, with which we are all familiar, you can't simply buy a new piece of hardware or hire an expert to get the system running again. If your system doesn't work, it is not likely to work for a long time.

Scope of the Problem

The problem exists for all of the mainframe computers and personal computers (PCs) in use throughout SSA. All the PC-based codes used in our regional offices and Program Service Centers must be examined. In addition, if any employees have written programs currently in use, the programs must be examined to determine if any changes are required. We will also need to determine when commercial software products which we use will be Year 2000-compliant.

Tools Available to Help

Although there is no automated solution, there are tools available that will help with this problem. We purchased one of these, the VIA/ALLIANCE software product from VIASOFT, in June 1995. This product helps identify dates in our computer system and tracks their flow as they are moved from field to field. We have already conducted training sessions on the use of this tool, and are in the process of using it to help us identify date fields in our programs. Use of this tool will also help with estimating resources needed to complete the project.

In addition to the newly-purchased software product, our repository which houses all of our software, ENDEVOR (Environment for Development and Operations), is equipped with its own scanning tool, which helps us focus on those areas in the code most likely to contain dates. While neither of these tools avoid the necessity of looking at every line, they will make the analysis phase of this project quicker and easier.

Other Areas Affected

The conversion to a system which can handle 21st century dates affects more than lines of computer code. Many forms currently in use have a preprinted "19" prefix in showing dates. Since preprinted forms require a long time to be revised, they must be changed as early as possible. Also, the computer screens that our employees see display only two-digit dates. Because the screens themselves are full, changing the dates to display a four-digit date would involve redesigning screens and the order in which data are displayed. As a result, we decided to continue to show two-digit years on the screens and use an algorithm to determine which century applies. We will redesign only those screens where the century cannot be determined.

Interagency Activities

I would like to briefly mention that the Office of Management and Budget asked SSA last year to lead interagency discussions designed to increase awareness of the Year 2000 issue and encourage a sense of urgency concerning the changes that will be needed. The Year 2000 Interagency Committee Chairperson is Kathleen Adams, Associate Commissioner for Software Design and Development, SSA. The Committee began meeting in November 1995 with only a few representatives attending, but the number of participants has grown to 30 Departments and Agencies, and the number grows with every meeting.

It is important to keep in mind that each organization must find solutions that meet its unique needs, and that there is no single approach that all agencies can employ. The purpose of the interagency committee is to discuss cross-cutting aspects of the problem, such as interagency data exchanges and availability of various vendor products. Furthermore, we believe that this group is meeting its objective to raise awareness of this issue and to encourage timely action. To this end, SSA and the Year 2000 Interagency Committee sponsored a Year 2000 conference for all Government agencies on May 2, 1996, in Washington, D.C. Approximately 500 people from Government agencies and private sector firms attended the conference and shared their experiences and approaches for addressing the Year 2000 problem in their organizations. Twenty-seven vendors that have Year 2000 solutions were also present to answer questions and distribute product literature.

Conclusion

In conclusion, Mr. Chairman, it would be impossible to overstate the importance of a smooth and timely transition of computer operations to accommodate the need to reflect 4-digit years as we enter the 21st century. There should be no question of what needs to be accomplished over the next several years, and no hesitancy in devoting the resources required to ensure timely completion of the task. I can assure you that SSA will continue to work to complete the project on time.

Chairwoman MORELLA. Ms. Adams, did you want to add anything or wait for the questioning?

Ms. ADAMS. No. I don't have anything.

Chairwoman MORELLA. Thank you. Thank you both for giving us kind of a synopsis of your testimony.

I wanted to ask Mr. de Jager, who seems to feel this is a really dire problem and that 65 percent of our businesses have absolutely ignored the reality of it coming about in 140 weekends. To put it in those terms, it just seems like it's around the corner.

Mr. DE JAGER. It is around the corner.

Chairwoman MORELLA. And, it is. It is. I wondered if you would comment on whether you agree or disagree with the Gartner Group's estimate of the cost to correct the Year 2000 problem within the federal government.

It said it's going to be $30 billion.

Mr. DE JAGER. The Gartner Group is a very well respected, highly respected, trend watcher in the industry. They are also very conservative in their estimates.

Chairwoman MORELLA. How do they arrive at a figure like that?

Mr. DE JAGER. There are several ways to do it. One of the simplest and most controversial is how many lines of code do you have in your organization. And, then you simply multiply the number of lines of code by $1.10 for every line.

There are other ways to do it. And, Capers Jones works on function points, which is a computer term to describe the complexity of a task.

And, you can rank a project based upon how many function points it contains. And, there is a cost associated with each function point.

Chairwoman MORELLA. Would you like to respond to the testimony that Mr. Mesterharm gave about how SSA is going to be ready by December 31st, 1998?

Mr. DE JAGER. I've had many conversations with SSA during the writing of many articles. And, I have nothing but the utmost respect for the heroic efforts that they are making.

There is only one problem. They are not getting support from their government.

They are currently working out of existing maintenance budgets, which makes this a practically impossible task. They need your support for proper Year 2000 budgets.

Otherwise, my prediction is they will most probably fail.

Chairwoman MORELLA. Aren't you glad I asked him that question, Mr. Mesterharm?

Mr. MESTERHARM. Yes, I've very glad you asked him that question.

[Laughter.]

Chairwoman MORELLA. But, they have been able to plan it so they can——

Mr. DE JAGER. They have done a tremendous amount of work.

Chairwoman MORELLA. (continuing) —do it, which indicates it can be done within a system. But, you are saying they will need that kind of continued support.

Mr. DE JAGER. They began many years ago. And, they are, indeed, at the forefront.

And, they've been doing the work for the last four or five years, actually longer. And, it's that proactiveness that has made it possible to achieve what they have achieved.

Chairwoman MORELLA. Is there a tendency sometimes to exaggerate the problem?

Mr. DE JAGER. We cannot exaggerate this problem beyond the reality. The bottom line is that when your program dies, if your organization depends upon that program, your organization stops.

Daily, in America, in the worldwide, we transfer $3 trillion through electronic funds transfer. That is done by computers.

If those computers go down, we stop moving $3 trillion worth of money everyday in our society. If we don't fix the programs, they will fail.

Chairwoman MORELLA. How do we get these 65 percent of businesses and governmental entities to recognize this and then to do something about it?

Would you like to comment on a possible solution?

Mr. DE JAGER. There are several opportunities. I've heard that the SEC is thinking about making it mandatory in the annual report for an audit line to state whether or not the organization that is publicly traded is addressing this issue. That would be one step.

Another step would be to make a very, very large announcement that this is a real problem. There are governors of certain states here in America that have a political sense, an official opinion, that this is a fraud.

Those need to be changed. This is not a fraud. You can test any computer system.

All you have to do is type zero, zero into the year and press the Enter key and watch what happens.

Chairwoman MORELLA. So, we can handle it. As a matter of fact, one of the groups that is going to be testifying on the second panel has been trying to bring it to the attention of the American public in terms of ads and whatever.

Let me ask our friends with SSA. Could you give the Subcommittee an estimate of the total cost of the agency's conversion? Maybe you would want to break it down by fiscal year, particularly as we look ahead to what Mr. de Jager said about the fact that you need to have our support.

So, you might want to then comment. Do you have sufficient funds?

How do you plan to obtain them if you don't?

Mr. MESTERHARM. To date, we've used approximately 100 man years in the effort.

Chairwoman MORELLA. Why do they use it "man" years? People years. I mean, man or woman years——

[Laughter.]

Mr. MESTERHARM. People years.

Chairwoman MORELLA. I'm kidding.

[Laughter.]

Mr. MESTERHARM. We have used 100 people years so far.

[Laughter.]

Mr. MESTERHARM. Our estimate is basically for the systems part of it. To convert our programs, it's somewhere around 300. We have approximately another 200 to go.

We have converted a little over 20 percent of the code but have done a lot of base line work in front of that. In addition to that, there are other areas that we need to involve in the agency that will have to do work.

So, the total agency-wide, I think, could be as high as 500 person years to accomplish it.

I think some of the points Mr. de Jager made are very much appropriate. I think there's the majority of businesses out there that do not understand the problem and do not take it seriously.

It is even worse in foreign countries—Europe. And, I've had several conversations with counterparts in Germany and the U.K. And, they are further behind than the States are in dealing with the issue. And, everyone deals with the same similar issue.

The United States is more automated than other countries are. So, it is a bigger issue here.

But, some of the issues are dependent on your environment, whether you are a date intensive type of shop. Social Security is.

The more date intensive you are, the more of an issue you have because you have more places to go in and look and change. And, it becomes more complicated sometimes the way the code is.

I think, you know, basically that the issue really is getting people to understand that there is a problem there. And, I think if proper focus is put on it that you can get the job done.

The question is whether people will understand, because that means there will be an affect on other projects and things that are going on. And, if you have the executive group that is in charge saying that they don't want to hear that, that they don't think it's that big of a problem, that it's only a two digit year, then you could have serious problems.

And, I think maybe that it is appropriate to overstate the situation somewhat because of the risk that is there and that you have to get the concern going. And, that was very hard.

I started in 1989 at Social Security to try and move this issue. It wasn't until a couple of years ago that I started to get other government agencies to start to listen to the issue.

And, I think the fall out from a possible situation is very dire in relation to what could happen.

Chairwoman MORELLA. Our own agencies, what kind of response are you getting from them?

Mr. MESTERHARM. I think initially at several conferences that I talked at, I think the majority felt that this was not that big an issue, that it was something they could solve in the last couple of years before the date changed. I think after a number of different symposiums and other things that that whole mood is changing and people are beginning to understand the problem now.

Chairwoman MORELLA. I'm going to yield time to my colleagues to ask questions and then get back to ask you a few others. But, you didn't really tell me whether you've got enough money to handle——

Mr. MESTERHARM. In relation to the money, I do not have a concern at this point in time.

Chairwoman MORELLA. Okay.

Mr. MESTERHARM. There is a possibility though that I would have to move resources from one effort to another as we get closer in. But, at this point in time, I do not see a problem.

Chairwoman MORELLA. What are the other countries going to do? If they are further behind than we are—

Mr. MESTERHARM. The U.K. and Germany and others, their general tendency, when they talk about this, is that the U.S. is really ahead in this and that there is some type of quick fix that we are going to come up with. And, that's not going to happen.

And, we've had people come over to try and figure out a way where they can generate interest over there and to get people involved.

Chairwoman MORELLA. I am going to now yield time to Mr. Tanner for any questions.

Mr. TANNER. Thank you, Madam Chairman. As I said earlier in my statement, I want to commend you all for your foresight and your attention to this matter.

And, I really have no doubt that given proper support from the Congress you will be able to fix your system. It seems to me the major problem facing this is not what people who are aware of the problem and who have the foresight and intellectual capacity to fix the problem are all about, but how in the hell are we going to coordinate all of these thousands of different programs from all over the country from people who report to you, who send in time sensitive, whether it be records, tax returns, whatever?

And, this, I would like to ask you what your thoughts on this, the complexity on bringing all this together. It's hardly, it seems to me, a way to fix a program, for you all to fix your own systems, and then nobody can talk to it, so to speak.

And, I have to use laymen's terms, because I am not educated in this field. Would you help me with that?

Mr. DE JAGER. One of the best things that you could do is to set a standard for departmental data transfers, that all dates will be of the form—four digits for the year, two digits for the month, two digits for the day. And, that is now the interdepartmental standard.

One of the things that we suffer under in the IS community is that we have so many standards. Just choose one.

And, no one has ever come forward and said, "No, this is it." And, it's maybe even passed into law, that if you are passing information from one organization to another that you will pass it in this format.

Now, that's heavy handed, but it may be the thing that is necessary, because you only have three and a half years left.

Mr. TANNER. I was thinking when ya'll were testifying, we have these hearings every so often involving NIST and the standards. And, we always go back to the reason for standards.

And, when the City of Baltimore caught on fire and they called all the fire departments in 1908, or whatever it was, and none of the hoses fit each other, this makes that fire look like a picnic, doesn't it?

[Laughter.]

Mr. TANNER. Do you think NIST has a role to play in this?

Mr. DE JAGER. I believe all your standards organizations have a role to play in this. What needs to happen are some government direction for some of the organizations to do this in a particular way.

Mr. TANNER. Do you all have any thoughts about it, Mr. Mesterharm?

Mr. MESTERHARM. Obviously, we have a lot of data exchanges with all the states. And, it is something that we are very concerned about, as well as a number of the agencies and departments who exchange data with us.

I do not know—I don't feel comfortable, that's for certain, that all the states and all the people that we exchange data with will have this situation fixed by the Year 2000. It is something that we are looking at, each one of the files we receive and whether that is data that would be put into our files and what type of computations we make with that data that comes in, to try and have some kind of a risk analysis and a fall back situation and do some kind of an audit in a timely fashion around that Year 2000 change to be able to determine whether or not we should accept that data anymore and whether we should update our files, because it would contaminate the information that we would have then at that point.

Mr. TANNER. Well, I know people are adverse to more law, more regulation and so forth. But, I hear you all describe an absolutely cataclysmic event occurring on January the 1st, 2000 if something is not done.

And, this may be an unfair question. But, in your opinion, is the size of this cataclysmic event, should it occur, such that we ought to at least consider some sort of public policy statement, a law or some regulation as to how people exchange data with you in the next year or so, so that we either, by force or by inducement in some way, force action that must be taken to avoid this calamity that could befall us?

In other words, is this so serious that we ought to consider actually the heavy hand of the law, as you referred to it?

Is it that serious?

Mr. DE JAGER. It is that serious and more. I, like you, do not like extra regulation.

But, if I need to communicate with you, I have to be communicating with you in a form that you can understand. If we are passing dates around three years from now the way we are passing them around today, that understanding is not going to happen.

You would actually make the task much simpler for many organizations if you chose a standard and said, "This is how we are doing it."

Mr. MESTERHARM. I agree with the fact that we need to have a standard that needs to be instituted. And, I think we need some kind of leverage behind that to make sure——

Mr. TANNER. You would recommend force of law?

Mr. DE JAGER. That's the only thing that this market understands sometimes. And, this may be the case where it's called for.

Mr. TANNER. All right. Thank you.

Chairwoman MORELLA. Mr. Calvert.

Mr. CALVERT. Thank you. We are a country of optimists. We always believe that problems are going to go away. And, in this case, obviously the problem is coming upon us very quickly.

You mentioned a quick fix. And, everyone probably is out there looking for that quick fix.

And, from Mr. de Jager's reaction, I suspect that quick fix may not be all that easy to find. So, the only way, from what my understanding here is, to take care of this problem is basically to have thousands of coders out there looking at every single line of data.

Is that what you were saying?

Mr. DE JAGER. There are some phenomenally smart and intelligent and ingenious programs out there that can help you with both the inventory and the analysis of everything that we have to do. When people speak about a quick fix, they literally mean something like "take two aspirins and call me in the morning and it will all be gone."

There is no way to wave this one away. You can't create a virus, for example, that will go into all your computer programs and make this thing go away.

The tools are incredibly smart. Anybody entering into this project without taking benefit of those tools is being negligent of resources, to say the least.

But, there is no magic bullet, silver bullet, that you will find. For one thing, there are two—at least 2,000 program languages worldwide. Many organizations don't have the source code for the programs that they are running. They only have what is called an object module, which has to be reverse engineered so you know what it's doing.

Most of these things—most applications are missing documentation that explain what the program is going to do, what it's supposed to do. And, it's very difficult for a programmer to go in and fix something if you don't know what it was doing in the first place.

So, there is no magic fix. But, there are lots of powerful tools. And, those tools should be used.

Mr. CALVERT. Well, is there any companies that recognize this problem that are starting up—your company obviously, Mr. de Jager—that go into corporations? I suspect if you are in business—at least, I would think you would—you are looking down the line.

Are some of our major companies—banking, as you mentioned earlier—hiring people to come in and take care of this problem or is this, for the most part, just totally being ignored?

Mr. DE JAGER. No. The banks and the insurance companies, a good solid, 75 percent to 80 percent of those organizations, are actively solving this thing. They are dealing with people's money. They know that if they are not dealing with this, they will be in trouble.

It's utility companies and government departments and retail stores that, for the most part, are ignoring it.

And, as to, are there people out there solving this? We've got five new companies springing up everyday. This is a growth industry, the likes of which we have never seen.

Mr. CALVERT. Thank you.

Chairwoman MORELLA. We've been joined by Mr. Gutknecht.

Mr. GUTKNECHT. Thank you, Madam Chairman. I just want to thank you for holding this hearing.

I must say that earlier when Mr. de Jager was talking, he said it's hard to get government officials to take this seriously. And, I found myself smiling. I mean, all of a sudden, you know, I think we are beginning to understand through this hearing the enormity of this problem.

I don't really have any questions for this panel. I hope we can hear from some folks on the next panel who maybe have some observations that will help us.

But, thank you very much for coming.

Chairwoman MORELLA. I find it very interesting to learn that the Governor of Nebraska has the two cents tax on tobacco extended and he has committed all the revenues to getting ready for the Year 2000. I just wondered if you wanted to comment on that?

Do you think that we are going to have to do that in more states or the federal government? Or, do you see this as a good sign?

It's kind of unique, isn't it? The State of Nebraska, only one body, you know. They don't have the two houses—

Mr. DE JAGER. I thought it was an incredibly ingenious and creative way to raise funds for this conversion. And, it's sort of fitting in a way.

People who smoke, they don't worry about the future.

[Laughter.]

Mr. DE JAGER. And, now they are going to have to pay the extra money to worry about this one. So, I was real pleased.

And, if I could add a comment on that, is I thought that that solution would have generated a tremendous amount of press coverage. And, one of the problems that we are wrestling with this whole problem is that the press has not been covering it.

And, if they do, they cover it under "Mad Computer Scientist Predicts the End of the World in the New Millennium," which, quite frankly, doesn't do anyone any good.

Chairwoman MORELLA. And, the press here, I'm sure, heard that.

Mr. DE JAGER. I'm sure they did. I can't imagine why I said it. [Laughter.]

Mr. CALVERT. Maybe if we explained that their paychecks won't clear on the Year 2000, all the reporters won't get a paycheck, they will understand.

Mr. DE JAGER. Or their pensions.

Chairwoman MORELLA. That's right, or the pensions. What about a tax on beer or alcoholic beverages? The same thing.

Mr. DE JAGER. Use whatever means you can to fund this.

Chairwoman MORELLA. What I'm curious about, if we undertake this problem or challenge seriously, both in governmental agencies as well as in the private sector, but if our compatriots overseas, whoever they may, whether it's below us, next to us, an ocean separates us, if they don't, isn't that going to screw things up?

I mean, this is not a problem just for the United States.

Mr. DE JAGER. No.

Chairwoman MORELLA. It's not a problem just for government. It's——

Mr. DE JAGER. It's a worldwide problem.

Chairwoman MORELLA. (continuing) —clearly a world problem.

Mr. DE JAGER. Yes. I think that's one of the reasons you asked a Canadian here.

Chairwoman MORELLA. Right.

Mr. DE JAGER. Worldwide, you need to solve the problem. You see, I am not really overly concerned about the people who are working on it.

Anybody who is working on this will ultimately muddle through. They will perform triage. They will do the absolute necessary things that they have to do to do business on a day to day basis.

But, if the U.S. starts making a real strong stance, every other country in the world will follow suit. They have to.

We are a global economy, the likes of which we have never been before. We all depend upon each other.

Chairwoman MORELLA. So, the market will handle that. Would you like to—before we go into our next panel, where you have given us a lot of the ammunition for asking questions with regard to standards, is there something that you would like to comment on?

Is there anything that we haven't asked you that you would like to mention?

Mr. TANNER. Yes.

Chairwoman MORELLA. I am going to recognize you also. I was just going to ask them if they wanted to say something and then turn to you, Mr. Tanner.

Mr. Tanner.

Mr. TANNER. Thank you, Madam Chairman. I understand that NIST has published a technical bulletin and that there are already national and international date format standards and so forth, but they are apparently voluntary.

And, from what I am hearing from you all, unfortunately that doesn't seem to be as widespread in its acceptance or in adherence to it as you perhaps think it should be. Is that a fair statement?

Mr. DE JAGER. That's a fair statement.

Mr. TANNER. Which brings me back to what I said awhile ago. We like to think the market can take its own measure and do the things that it needs to do.

But, if it doesn't, in this case, is the risk involved worth us trying to pass some sort of law to guide commercial activity in exchange of information in this area between now and the Year 2000?

Mr. DE JAGER. If you kept the law to only the data that is transferred from one organization to another, you would gain some acceptance in the marketplace for that. If you tried to pass a law that determines how I store data in my own computer, then——

Mr. TANNER. No. I'm just talking about the exchange.

Mr. DE JAGER. Okay.

Mr. TANNER. So, everybody is talking the same language to each other in terms of computers.

Mr. DE JAGER. That would be a tremendous help to the industry.

Mr. TANNER. What is the level of compliance—do any of you know—with the NIST advisory? Is there any way to know?

Mr. MESTERHARM. It was just published.

Mr. TANNER. Thank you.

Chairwoman MORELLA. Mr. Calvert.

Mr. CALVERT. No further questions.

Chairwoman MORELLA. Okay. Mr. Gutknecht?

Mr. GUTKNECHT. No questions.

Chairwoman MORELLA. It just occurred to me finally, in thinking about our federal agencies, DOD, the Department of Defense, what are they doing to ensure that our allies have converted their weapon systems for the Year 2000?

What if they didn't do any conversions? What would happen?

I mean, is the Interagency Committee looking at that?

Mr. MESTERHARM. The Interagency Committee has DOD membership on it. Whether there has been a look at the allied or not, I have no——

Ms. ADAMS. Basically, the Interagency Committee is providing a forum to exchange information and learn as we go along and address issues like standards and contract language that we can share or put into contracts to protect us so that we don't buy anymore software that isn't compliant, because as we sit here, people are writing software across the country that is not compliant, and saying to the vending community that as of a certain date you are not going to get on the GSA schedules if your software isn't compliant, because a lot of the commercial, off-the-shelf software that is out there today is not Year 2000 compliant.

But, it is really not the job of the Committee to look into a specific agency and the challenges that it faces in making these changes. That would be up to the agency head and the CIO. And, I'm sure that Defense could answer.

Chairwoman MORELLA. Are all the agencies involved?

Ms. ADAMS. We are up to 31 departments and agencies on the Committee. So, we've just about got everybody covered at this point.

Chairwoman MORELLA. Okay. I want to thank the panel. You've been very informative.

Mr. de Jager, thank you for coming from Canada. We appreciated that and your expertise.

And, I want to thank the Social Security Administration, Mr. Mesterharm and Ms. Adams, for being in the forefront in terms of being ready for this. When did they first start, you know, doing the four digit?

Mr. MESTERHARM. 1989 is when we started.

Chairwoman MORELLA. Okay. Was it—is that kind of the date from that point on that the four digits were recognized in computers?

Mr. MESTERHARM. Well, actually there wasn't any standard that was in place. If you go back, from a historical perspective, when computers started and you had card equipment, you had a one digit standard and they would only carry the last digit of the year because you had an 80-column card and you were trying to save space.

Then, as computers grew and everything, you got tape storage. Then, they moved to a two digit standard. And, again it was to save memory and to save storage space.

In the early 80s is when a four digit standard should have been implemented. And, that did not take place. And, that's kind of why we are where we are now.

Chairwoman MORELLA. I want to thank you very much for excellent testimony and responses to our questions. I hope you will stay involved with us as we try to give some kind of direction to Congress. Thank you.

Our second panel, we have four people on the second panel. Dr. Robert Hebner, who is the Acting Deputy Director of the National Institutes of Standards and Technology in Gaithersburg, Maryland; Mr. Barry Ingram, Chief Technical Officer of EDS, Government Services Group in Herndon, Virginia, and they are the ones that have the ad that I have referred to; Ms. Barbara McDuffie, Program Director System 390, Solution Provider Program, IBM, in Poughkeepsie, New York; and, Mr. Marc Sokol, Vice President of Advanced Technologies, Computer Associates, Islandia, New York.

Maybe we will just do it in the order in which you are sitting. Dr. Hebner, we will start off with you, sir.

STATEMENT OF ROBERT HEBNER, ACTING DEPUTY DIRECTOR, NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY, GAITHERSBURG, MARYLAND.

Mr. HEBNER. Madam Chairwoman and members of the Subcommittee, I thank you for inviting me to come and discuss with you briefly this afternoon the Year 2000 computer problem. When my colleagues, friends and neighbors ask me if this is really a problem, I have a very simple anecdote that I give them to help them understand it. And, it serves as a basis for what I have in my written testimony.

I ask them to just think back to the last time we changed to Daylight Savings Time and they had to go around their house changing all their clocks.

It's a very simple change. You move by one hour.

Inevitably, they smile, because they remember that there was a clock that they forgot to change and a light came on when they didn't want it to. Or, there was a clock they had forgotten how to change, because each clock in their house is different and you have to remember how to change it. And, the change of a single hour is a very simple idea, but it causes a bit of consternation every six months.

The same—there are some lessons to be learned from that as we go into the Year 2000. Number one, we've talked very much about the use of voluntary standards.

And, the key standard that NIST has promulgated was the Federal Information Processing Standard, which adopted the voluntary standard that had in it the four digit date code. We adopted the Federal Information Processing Standard back in 1988.

And, a couple of months ago, we strongly recommended that the federal agencies use the four digit code.

The other aspect, which is important to recall, is that each of the systems is individual. And, there is not a standard technique for fixing someone's system.

This has to be an individual activity, looking at the system. There are private sector efforts in industry to come up with common techniques to look at where there is commonality.

But, this is not a standards issue. So, we are focusing on what we think are the measurements of standard issue and are listening very carefully to what is being said here today.

Finally, for our internal systems, we are in a fortunate position that the only date intensive system that we had was our Management Information System, which takes care of things like procurement, personnel and purchasing. And, the Department of Commerce started several years ago to completely revamp that entire system.

And, we expect to have the new system on line within the next year or so. And, it fixes the Year 2000 problem.

So, we are upgrading all of our systems to do that as part of our normal improvement of the Department's information management.

Thank you very much. And, I welcome your questions.

[The prepared statement of Mr. Hebner follows:]

STATEMENT OF ROBERT HEBNER

ACTING DEPUTY DIRECTOR

NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY

BEFORE THE

SCIENCE SUBCOMMITTEE ON TECHNOLOGY

U.S. HOUSE OF REPRESENTATIVES

MAY 14, 1996

Madam Chairwoman and Members of the Subcommittee, I am Robert Hebner, Acting Deputy Director of the National Institute of Standards and Technology (NIST). The Department of Commerce has a key role to play in addressing technology, information and global competition. The Department's primary mission is to ensure economic opportunity and a high standard of living for all Americans through economic growth and job creation, promotion of trade, and advances in technology.

I am pleased to be here with you today to discuss the "Year 2000" computer problem and activities at the National Institute of Standards and Technology in response to this problem.

As you have already heard, the year 2000 problem is due to computer systems recognizing a two digit year of "00" as 1900. While that may appear to be a trivial problem to fix, it is not. Computers follow the logic they are designed or programmed to follow. If that logic was developed with the assumption that "00" meant 1900, then that logic itself must be re-thought to include the possibility that "00" means 2000, not 1900. In complex systems rethinking the logic is a time-consuming, difficult task that requires technicians to make delicate fixes to programmed code.

While the actual solution for any particular system will require hands-on work by technicians to fix, there are some things that we can do government-wide to facilitate that work.

NIST ACTIONS AND PLANS

The first is to raise awareness of the problem and assure senior management support for solving it. In that regard, hearings such as this, and recent communications to senior agency managers by the Office of Management and Budget help. So do articles such as the recent *Government Executive* cover story.

The second way we can help is through sharing of information. In that regard I note the leadership of the Social Security Administration in the inter-agency working group to share information. Here we have been able to contribute significantly. First, by adopting a voluntary industry standard in Federal Information Processing Standard (FIPS) 4-1, "Representation for Calendar Date and Ordinal Date for Information Interchange," NIST has advised agencies on the use of standards in implementing Year 2000 solutions. On March 25, 1996, NIST began recommending, rather than simply listing as an option, the use of 4-digit years in data that is electronically transmitted among federal agencies. That is important because many federal systems inter-operate across agencies. As each agency implements its own technical solution to the year 2000, they can now design to a standard way of sending and receiving dates from other agencies. Additionally, NIST published a technical bulletin in March providing information to agencies and industry on acquiring help in solving the Year 2000 problem. A World Wide Web page concerning Year 2000 issues has been implemented in response to government and industry requests for assistance and guidance. Our communications reach a different, technical audience than many others do. Since this problem potentially can affect virtually all aspects of the Federal community, it is important to discuss it in as many venues as possible.

NIST believes the Year 2000 issue is a computer programming and resources problem that can be solved with current technology, but it requires immediate management attention. There is no across-the-board solution and no need for additional standards. There already are national and international date *format* standards. A date *processing* standard would not be meaningful due to the numerous individual agency requirements within specific applications and systems. A "one-size-fits-all" standard on date processing probably would not work, nor could such a standard be developed in a timely manner. Date processing routines are already provided in many programming languages and software development environments.

The information technology industry already is responding with new and updated products and methods to assist agencies in converting their software and databases. The General Services Administration is busy developing the text needed to define requirements for specifying Year 2000 compliance in procurements. Vendors and industry associations are studying the problem and possible solutions, refining their offerings based on new developments and the results of these studies.

The most reasonable solution is to attack the problem one step at a time. Here is a basic approach to a solution:

- Select a product to assist in managing the inventory of software and databases involved. Select one or more products to assist in analyzing the software and estimating the extent of the problem. Some of these products also will modify the software and data automatically, but cannot do so for every case. (Some computations are date-related, but cannot be determined from the source code. In such cases, an individual must analyze the source code line by line.)
- Inventory applications, libraries, databases, extraneous files, documentation, and other items that have importance within specific systems. Identify who is responsible for each item.
- Analyze the applications and data. Estimate modifying the source code alone to change those locations that perform date computations and logic operations based on dates. Perform a second estimation that includes modifying databases and all source code that references data fields and all source code affected in the first estimate. If there is an insignificant difference between the two estimates, the recommended course of action is to modify both the databases and

the source code. It may be less expensive in the short run to modify only the source code, but more expensive in the long run if maintenance problems crop up over time due to the date processing fixes.

- Assemble a team of programmers, application experts, database designers, and project management based on the overall system requirements. Once estimates are known, the number of personnel required can be determined, particularly in view of the automated tools selected for use.
- Modify the system. Three major options are: a) modify the source code to manipulate and perform computations on dates with century digits included; b) use a sliding window time frame to determine date context for computations; and c) incorporate packed date fields and use specialized subroutines for performing the computations. All three of these are expensive and may lead to further maintenance problems in the long run.
- Test the modifications. Allow 40-50 percent of the overall project resources for testing, even more if the database is modified. This includes testing documentation to ensure that directions are correct and correspond to the changes made.

There is one other facet of sharing information where we can play an important role: fostering communication of technical ways to fix the problem. As I noted earlier, the solution to this problem in any organization will require technicians to do hands-on fixes. Making a fix will involve understanding the different possible ways to craft a solution, then selecting the best one and properly implementing it. At NIST we will foster communications regarding the different ways to craft a solution. Earlier, I mentioned our technical bulletin on the year 2000. That is one example of such communication. We can also raise this issue in the various professional organizations with which we routinely deal. After all, unless we assure that there is the technical ability to actually fix the problem, awareness by management will only lead to frustration.

At NIST, we have analyzed the impact that the Year 2000 problem could have on the systems we use to do our own business, and the impact will not be significant. Sixty percent of the existing computing *systems* will be replaced with a new Commerce Administrative Management System that already integrates 4-digit year processing. The remaining systems will require some modifications. Personal computers and other hardware that show date problems will be replaced through the normal computer upgrades between now and the end of the century.

That concludes my testimony. I would be pleased to answer any questions you may have.

DR. ROBERT E. HEBNER

Acting Deputy Director

National Institute of Standards and Technology

Robert E. Hebner received his Ph.D. in physics from the University of Missouri in 1971. He joined what was then the National Bureau of Standards as a presidential intern. Since that time he has advanced through increasingly responsible positions and is currently Acting Deputy Director of NIST.

His activities with other agencies have included assignments at the Office of Management and Budget, at Sandia National Laboratories, and most recently, at the Advanced Research Projects Agency of the Department of Defense. He has served as a member of the Advisory Committees for two NATO Advanced Study Institutes and has participated in the organization of many national and international conferences. Dr. Hebner was a member of the review committee for U.S. Department of Energy's Office of Energy Storage and Conservation in 1986 and on an NSF Panel for Engineering Research Equipment Grants. In addition, he served on the National Research Council's Committee to Conduct an Assessment on Establishing Underwater Arcs and Flames. He is a member of a Federal interagency committee to coordinate the U.S. research and communication program on the potential health effects of electric and magnetic fields. He was the principal investigator and the government's key witness in a celebrated court case involving an alleged novel approach to the efficient generation of electricity.

Throughout his career, Dr. Hebner has been active in technical activities having authored or coauthored more than fifty technical papers and reports. He is a past president of the Dielectrics and Electrical Insulation Society of the Institute of Electrical and Electronics Engineers. In addition, he has served on numerous technical committees which develop voluntary standards for the electric utility industry.

Dr. Hebner is a member of the American Association for the Advancement of Science, the American Physical Society, and is a fellow of the Institute of Electrical and Electronics Engineers.

The awards that he has received include the U.S. Department of Commerce Silver and Bronze Medals and the 1990 Harry Diamond Memorial Award given annually by the Institute of Electrical and Electronic Engineers for outstanding technical contributions in the field of government service. The award cites Dr. Hebner's contribution to the development of electrical measurements and standards for the electrical utility industry.

Chairwoman MORELLA. I think you are going to be getting some.

Mr. HEBNER. I think so, too.

[Laughter.]

Chairwoman MORELLA. Mr. Ingram.

STATEMENT OF BARRY INGRAM, CHIEF TECHNICAL OFFICER AND VICE PRESIDENT, TECHNICAL SERVICES, GOVERNMENT SERVICES GROUP, EDS, HERNDON, VIRGINIA

Mr. INGRAM. Thank you. I am Barry Ingram. I have been involved in the information technology field for over 32 years, so I am one of the people that remembers the punch cards in the old days.

I want to thank you and your staff for the opportunity to be here today and for soliciting our views on the Year 2000 problem.

The current estimate of $30 billion to solve the Year 2000 problem for the federal government has not been verified. And, until the impact analyses are completed, the government will not know the scope of the problem or the most effective solutions.

By participating in today's hearing, I hope to assist in providing some guidelines for preparing an impact analysis. And, I also hope to underscore the overriding importance of the management process in responding to the Year 2000 challenges.

An impact analysis incorporates a mission plan, the information technology plan and the results of a date utilization investigation or an assessment to perform a cost versus benefits versus risk study to select the appropriate action for bringing each system into Year 2000 compliance. The methodologies for making systems Year 2000 compliant cover the gamut.

A sample of these are a brute-force conversion model, selective re-engineering or conversion and interfacing and then, finally, total reengineering and redevelopment. There's disadvantages and advantages to each of these scenarios, depending on the particular application and environment.

Systems that, for whatever reason, have significant needs for upgrades and modernization might best be suited for either selective reengineering or redevelopment. Those systems, however, that still have a reasonable life expectancy, have been well maintained in a controlled environment and are performing well and providing some business value may be candidates for a conversion scenario.

The important issue in this is that the determination should be a conscious, educated one rather than just one looking at a single criteria such as cost.

While the Year 2000 problem will most assuredly present a significant drain on our skilled and scarce technical resources, the overall management aspects of the problem are staggering. Rarely do we imagine projects of this magnitude where every line of code, every piece of data, every report, every terminal screen and every form must be inspected, analyzed and assessed for the impact.

Following that, a project to upgrade and test these systems must be initiated, where again the management aspects of assuring completion of the effort on time and within budget and to the required operational capabilities are looming over us. All these Year 2000 activities must still happen while normal development maintenance efforts proceed.

It's imperative that the federal government and industry act now. The greater the delay, the fewer the resources that will be available to work on the transitions.

Those agencies and corporations that begin now will have access to the best resources and will be able to complete the Year 2000 projects at a lower cost than those that choose to wait. The later entries in the effort may not even have access to the best and most capable resources and will most likely pay a premium for the scarce personnel available at that time.

Many of those conversions are so large or those projects are so large that if they wait they may not even have resources to get it done.

Classical project management disciplines allow for replanning by adjusting scope, adjusting resources or adjusting time constraints or even a combination of the three. With the Year 2000 effort, however, scope adjustment is not an option. Schedule adjustment is out of the question. And, that leaves only one option, and that's project management practices that effectively manage the resources.

We must learn from each other and remain focused on the final results needed. I believe industry can contribute to the government's solution by providing some skilled resources and experiences, whether through outsourcing or current contract vehicles or providing tools and methodologies or providing consulting services.

But, the time to address the issue is now. We cannot afford to wait any longer.

The Year 2000 will occur on time. And, we must be ready. We are in a race against time in preparing for the Year 2000.

Thank you for inviting me to today's hearing.

[The prepared statement of Mr. Ingram follows:]

TESTIMONY OF BARRY INGRAM

EDS CORPORATION

CHIEF TECHNOLOGY OFFICER AND VICE-PRESIDENT TECHNICAL SERVICES

GOVERNMENT SERVICES GROUP

BEFORE THE SUBCOMMITTEE ON TECHNOLOGY

HOUSE SCIENCE COMMITTEE

MAY 14, 1996

INTRODUCTION

Good afternoon. I am Barry Ingram, Chief Technology Officer and Vice-President of Technical Services, EDS Government Services Group. I have been involved in the information technology field for over 32 years. I want to thank you and your staff for the opportunity to be here today and for soliciting EDS' views on Federal implications of the Year 2000 date problem. EDS has operations across the United States and in 41 countries around the world. We have customers in a variety of industries, including health care, banking, manufacturing, transportation, retail, communications, and government.

I commend you for conducting this hearing that will add to the public awareness of the need to begin the largest computer initiative in history—The Year 2000 program. If not attended to, this situation could lead to the failure of computer systems on a global basis.

The current estimate of $30 billion dollars to solve the Year 2000 problem for the Federal Government has not been verified and, until the agency Impact Analyses are completed, the Government will not know the full scope of the problem or the most effective solutions for preparing an Impact Analysis. By participating in today's hearing, I hope to assist by providing guidelines for agencies to consider as they prepare for responding to it. An Impact Analysis incorporates the Mission Plan, the Information Technology Plan, and the results of the date utilization investigation, or assessment, to perform a study of costs/benefits/risks in order to select the appropriate action for bringing each system into Year 2000 compliance. I also hope to underscore the overriding importance of the management process in responding to the Year 2000 challenges for the Federal Government.

The Year 2000 Problem

Date information plays a major role in almost all computer applications developed over the last thirty years. Computer systems utilize date information through performing complex calculations, sorting information and storing information for later retrieval and analysis. The de facto standard for storing and processing date information has, for a variety of reasons, been to represent the year in a two-digit format. Many of these reasons have already been addressed: excessive cost, memory and storage limitations, processor speed constraints, and others. Although this method worked for most date calculations, as we approach the Year 2000 and encounter date calculations spanning the century, systems are beginning to fail, and more will fail at an increasing rate until we address and resolve the issue.

It is imperative the Federal Government and industry act now to resolve this issue. The greater the delay, the fewer the resources that will be available to work on the transitions. Those entities—agencies and corporations alike—that begin now will have access to the best resources and will be able to complete their Year 2000 projects at a lower cost than those that choose to wait. The later entries in the effort may not have access to the best and the most highly capable resources, and will most likely pay a significant premium price for the scarce experienced personnel available at that time. Many of the required conversions or reengineering projects are so large that, if one waits, one may not have the time to make the conversions—no matter how large the available resource pool or how much one is willing to spend.

Impact

The impact on today's computer systems is felt throughout the entire system hierarchy, from the program level all the way to the global level. The major considerations at each of these levels are:

Program Level

- The application of standard naming conventions has not been uniformly applied. Many systems have been developed with no standards at all and, without this guidance, programmers have shown great creativity in naming date fields. This increases the difficulty of applying automated tools to locate the dates and associated fields within programs.
- Installations have had to deal with multiple date formats, caused by lack of organization or industry-wide date formatting standards. This lack of standards and development and maintenance of the programs by many individuals over time has led to a proliferation of formats.
- Hard-coded literals, or fields with specific values, such as "19" to represent a century to be printed on forms, are commonly used.
- The utilization of date information within data base keys may have an impact on the structure of the data base. Modification of this information may also have an effect on the overall program performance because of restructuring.
- Dates appear on application screens, reports, and forms. Should there be a display space shortage on a screen, it may need to be redesigned to allow enough room to contain a four digit date field.

System Level

- System software, or that software providing the basic interface between the hardware and application software as well as the overall control environment, must be reviewed to determine if the vendor will be updating the current version to handle the dates correctly. If not, this will cause an additional step in converting to the Year 2000 compliant release of the system software.
- Systems developed utilizing multiple platforms may require multiple sets of conversion software to correct the code on all platforms.
- Commercial-Off-The-Shelf (COTS) software users must analyze vendor plans for which versions of their software are compliant and develop plans for upgrading to those versions.
- Even the newer client/server system architectures will require analysis, since many of these were also written without consistent date standards or were required to interface with pre-existing databases, archives, or other systems.

Enterprise Level

- Today's computer systems rely on the utilization of shared data bases across multiple applications. Therefore, any changes must also be coordinated across all these environments.
- The demand for on-going maintenance on today's systems will continue through the life of the Year 2000 update process. Although it is not normally recommended to make more than one type of change at one time, it may be necessary. In this case, the configuration management and testing processes become even more important and complex.
- Current systems development must be managed to ensure the practice of utilizing a two-digit date does not continue. Since the two-digit year has been the standard for so long, it may continue to be used in systems currently being developed. Again, the establishment and enforcement of consistent standards will help.

Global Level

- Data interchange between organizations outside of one's control must be coordinated to insure interoperability of systems.
- The demand for resources will continue to grow at the time when they are least available. Pools of resources around the world will be utilized to refurbish or reengineer systems.

Year 2000 Policy Guidelines

The Year 2000 solutions extend beyond analyzing software code, upgrading hardware, and restructuring data records, chiefly because this is not simply an information technology issue. The Year 2000 will have an impact on entire enterprises—from integration of internal systems to data exchanged with other entities; from domestic to international; from data systems that process information to process control systems that manage facilities; and from internal relationships to external relationships with our citizens and government agencies.

The solution strategy that an enterprise undertakes will be significant in terms of direct expense and the cost of lost opportunity. The system changes associated with the Year 2000 are extensive, and any time or resources utilized on the Year 2000 issue will be taken from other current projects or opportunities. The system changes associated with the Year 2000 are extensive. However, two keys to minimizing risk and maximizing value include establishing an enterprise strategy and executing it with discipline. Given the breadth of the systems that will be affected, intensive management control and involvement is a key success factor.

The following elements must be considered for generating policy guidelines for an effective Year 2000 program:

- There is no "Magic Silver Bullet."
- "Year 2000 compliance" must be well understood.
- Executive level support is essential.
- Strong project management techniques must be implemented.
- Incorporation of a consistent methodology for systems life cycle development and maintenance is paramount for success.
- Implementation and use of an effective configuration management process will provide the control to manage change.
- Utilization of an automated tool suite can provide significant productivity improvement on large systems.

- Outside organizations, both Governmental and private industry, can provide significant assistance and alternative solutions for accomplishing the mission.

Now, I will address each of these elements in more detail.

No Magic "Silver Bullet"

No single automated tool exists can be applied to the enterprise's entire portfolio to obtain Year 2000 compliance. The mix of platforms and languages may require a mixture of solutions. Each of the currently commercially available Year 2000 software assessment and conversion packages was developed for a particular set of software languages. Since COBOL is the language used in approximately 60% of the legacy systems, Year 2000 software developers addressed this language first. Fourth generation and other more modern languages are used far less extensively in the market and will most likely not be featured in their own respective conversion software. In addition, according to current findings, some enterprises are discovering that between one percent and five percent of the source code for their applications is not available (missing, lost, unidentifiable, unknown). In cases where the code has been used without change for many years and where there has been considerable staff turnover, the "unavailable" code can approach 20-30 percent. This code must be replaced through either reverse engineering or redevelopment efforts, both of which are time and labor intensive.

Year 2000 Compliance

Year 2000 compliance does not merely consist of converting date fields from two to four digits. To be deemed compliant, a system must be able to process all date information for duration calculations, comparing, and sequencing—including dates in the 20th and 21st centuries. This also includes the correct determination of leap years, the elimination of special codes sometimes stored in date fields, and hard-coded literals. Several definitions of this compliance state are being developed and recommended at this time.

Executive Level Support

Because of the global nature of the Year 2000 project, executive level support and sponsorship is required. Top level mission and technology decisions will be required. Without the direct involvement of executive level management, the coordination and assignment of the correct resources to address this issue will not occur. In addition to adjudicating internal conflicts that will occur during the solution development cycle, management must assist in the external coordination and cooperation with other agencies for information exchange. Without a common standard across the Government, not all agencies will be resolving their Year 2000 problems with the same sets of rules, and inconsistencies may occur.

Project Management

The single most important component of the Year 2000 efforts is **project management.** Managing the resolution of the Year 2000 exposure is a large, complex project requiring discipline over an extended time frame—quite likely the largest ever attempted. It requires implementing large, technology-intensive, time-sensitive, mission critical, global projects, with a completion date that cannot be extended. Therefore, it is essential to establish a core management team whose intent will be to see the project through from inception to completion. The eight major project management functions are:

- *Scope/Requirements Management*—Leads the identification of requirements and deliverables and establishes an understanding and agreement with executive level management about the project requirements and deliverables.
- *Quality Management*—Analyzes and monitors compliance to the project's goals and standards in accordance with the expectations of the enterprise.
- *Resource Management*—Creates the strategy and determines the procedures for timely, cost-effective acquisition, use, and allocation of resources to provide quality products and services.
- *Schedule Management*—Creates and maintains the project schedule to encompass all events, deliverables, and resources required within the scope of the project.
- *Risk Management*—Assesses the risk factors of a project and identifies specific risk items along with the contingency plans to deal with them.
- *Communication Management*—Determines what type of messages to send, to whom they should be sent, when to send them and how to translate them so all project participants are kept aware of their tasks and the progress of the

project. This also includes communication to all other agencies potentially impacted by the changes.
- *Contract Management*—Identifies and monitors adherence to the legally binding requirements of the enterprise's vendor contracts.
- *Financial Management*—Establishes the financial infrastructure to support the estimating, forecasting, budgeting, and tracking of the project economics.

Consistent Methodology

The Year 2000 project will require a well-defined set of procedures and tools for assessing, improving, and transforming legacy systems. The three key components of the Year 2000 methodology are assessment, improvement, and transformation. Assessment is a situation and planning process that determines the appropriate maintenance and software performance engineering strategies for a given application to meet future requirements. Improvements use source code analysis and modification technologies to streamline maintenance and prepare systems for transformation. Transformation facilitates migration, reverse engineering, selective redevelopment, and total redevelopment of legacy systems.

Configuration Management

Configuration Management (CM) provides the control mechanisms necessary to manage changes to software. CM leads to improved security, integrity, and audit capability and assists in performing the following functions:

- *Inventory Management*—The classification, storage, and control of software components, such as programs and documentation, and the relationship among these components.
- *Change control*—The authorization, execution and recording of changes to software components.
- *Process control*—The transformation of software components from one state to another, such as compiling source code to an object module.
- *Migration*—The movement of software components to one or more destinations.

Automated Tool Suite

Most installations contain sufficient source code to make the decision to utilize automated tool sets mandatory. One can expect that at least 50% of the code requiring change can be handled with tools and, in some cases, the conversion could not be completed without this automated capability. The decision to include automated tools is driven by the platforms and languages to be processed. The current commercially available date scanning and correction software are provided for both the mainframe and workstation environments. The key decision of which platform to use for the conversion is based on the functionality of the packages and the availability of hardware resources. The advantage of a mainframe is that the source can be processed in its native libraries. Packages running on the personal computer have the potential of offering greater availability and cost savings. However, one must download the source to those platforms and ensure that control is not lost on the source.

A complete tool suite is composed of minimally the following types of software:

- *Program Management*—This is most likely the largest maintenance project an organization will be asked to complete. Control, accurate reporting, and resource management are critical throughout the process.
- *Planning, Scheduling, and Estimating*—Given the size, complexity, and duration of a Year 2000 project, it will need regular reevaluation and adjustment to meet the changing needs of an organization.
- *Configuration Management*—The Year 2000 Project will occur in parallel with ongoing maintenance and system development requirements, making program version control mandatory.
- *Scanning and Correction*—The current commercially available tools support the commonly used languages. It may be necessary to develop special software to assist in the non-standard languages.
- *Database Scanning and Correction*—Software tools for scanning and reformatting databases and files need to provide for verification of changes and integrity of the revised databases. Based upon the specific database architecture, some files may require offloading, modification, and then reloading. This process will require that adequate short-term additional storage capability also be in place.
- *Testing*—Because of the special nature of the Year 2000 testing requirements, it will be necessary to include software that controls the system and application

calendars in order to test for future conditions in addition tO the normal testing software.

Outside Assistance

The decision to utilize outside assistance can have a significant impact on the final results of a Year 2000 project. Although the upgrading of legacy systems for year 2000 compliance is basically a system reengineering project, the volume of changes and the requirement to correct the entire portfolio at the same time everyone else in the agency is making similar changes adds a degree of complexity not previously confronted. In addition, it must be underscored that *the deadline will not change.* Nor will the scope of the project. The only thing one can control is how to accomplish the project. Year 2000 work is an emerging technology, and new tools and procedures are being developed every day. The best practices to date indicate that a combination of resources produces the best results and decreases the learning curve and risks. Each organization must analyze its resources and the current demands on those resources to determine where outside resources can best be applied. The following sources should be considered:

- Outsourcing Contractors
- Other Government Agencies
- Internet/Intranets—The Internet is already heavily used to communicate ideas and solutions (and frustrations)
- Professional and Industry Associations
- Seminars
- Vendors

Impact Analysis

As earlier addressed, the Impact Analysis is the final determination of analyzing exposure to date problems, information technology status, and mission support requirements. The following points should be considered in preparing a Year 2000 plan to develop the Impact Analysis:

Year 2000 Project Plans

Just as there is no magic solution for Year 2000 software packages, the same is true for Project Plans. Each Project Plan must be tailored to the individual organization and operating environment. The key to success is to correctly identify organizational strengths and weaknesses, then build a team to tailor the Project Plan to address their situation. The following covers key issues to be considered in building a Project Plan:

- **Assessment**

  The first question to be answered in the assessment phase is—what is the exposure to the Year 2000 problem? In order to determine the potential risk it is necessary to establish a baseline inventory of source components. This baseline then must be analyzed for date utilization. This analysis can be partially automated for mainline computer languages, but it will require a large manual effort for non-mainline languages. This analysis requires locating all references to date data and determining the function of the references. With this information, several of the automated Year 2000 software packages will produce an estimate to convert the source code to a four-digit format. This is **not** the impact assessment. This estimate provides the costs of performing a straight conversion on all date fields. Some date fieldss will not require conversion to a four-digit format. For example, dates that appear on screens, forms, and reports need only be changed if required for a clear understanding. An Impact Analysis includes a costs/benefits/risk study to determine the appropriate action for each respective system.

- **Analysis and Planning**

  The second question to consider is, of all the options available for upgrading the system, which is best for a particular organization. This requires an analysis that considers existing Mission Plans, existing Information Technology Plans, and current and projected resource capabilities and utilization.

  Rather than advocate across-the-board changes to correct date limitations, one must examine the existing systems, rate their potential for risk, and then take action. Depending on the ratio of risk-to-return, one may choose to correct the problem, replace or retire the system, or transform a high-maintenance system into one that performs better and is easier and less expensive to maintain. Processes to be considered are:

- Business Process Reengineering (BPR)

- Business Information Planning (BIP)
- Commercial Off-the-Shelf Software (COTS)
- Innovative Technology
- Straight Date Field Conversion
- Data Base and File Conversions

- **Solution Execution**

In most instances the team should conduct a pilot on a representative sample of the computing environment. This is done to validate the tools, processes, standards, and methodologies in place.

- **Testing**

Testing costs will need to be included in the Impact Analysis estimate. Testing for the Year 2000 is estimated to comprise as much as half of the overall project costs. Regression testing in addition to date processing validation will have to be supported by special testing software that allows simulating Year 2000 processing dates.

Critical Success Factors

- Year 2000 project estimates must be based upon an understanding of the software size and relationship to resource productivity rate.
- Project management discipline must focus on effective resource management. The scope is fixed, the time is fixed, and whether an organization can meet the Year 2000 challenge will be determined by its ability to size the problem accurately, acquire the necessary resources, and effectively manage the resources.
- Project tracking and control will require periodic status that includes not just schedule progress, but progress in terms of resource efficiency.

Additional Benefits

While this testimony has centered on the development of a Year 2000 project plan and the importance of the management process in this effort, there are other significant gains from a well-conceived Year 2000 effort. These are:

- Streamlined Project Management Process, maybe even institutionalized
- Improved Business and Information Technologies Practice
- More complete systems and program documentation
- More complete inventory of hardware and software assets
- Potentially, upgraded systems software, based on the upgrade approach taken
- Identification and verification or replacement of all source code
- Increased knowledge of the enterprise and information available

Closing

Methodologies for making systems Year 2000 compliant cover the full gamut. A sample of includes:

- Brute-force system conversion
- Selective reengineering/conversion/interfacing
- Total reengineering and redevelopment

There are advantages and disadvantages to each scenario, depending on the particular application and environment. Systems that, for whatever reason, have significant requirements for upgrades and modernization might best be suited for either selective reengineering or redevelopment. Those systems, however, that still have a reasonable life expectancy, have been maintained in a well-controlled environment, and are performing well in providing business value may be candidates for the conversion scenario. The important issue is that this determination should be a conscious, educated one, rather than just looking at a single criteria, such as cost.

While the Year 2000 problem will most assuredly present a significant drain on skilled and scarce technical resources, the overall management aspects of the problem are staggering. Rarely do we contemplate projects of this magnitude, where every line of code, every piece of data, every report, every terminal screen, and every form must be inspected, analyzed, and assessed for impact. Then, a project to upgrade and test these systems must be definedd and initiated, where, again, the management aspects of ensuring the completion of the effort on time, within budget, and to the required operational capabilities are looming over us.

Classical project management disciplines allow for re-planing due to poor performance by adjusting scope, adjusting resources, and adjusting time constraints, or a combination of the three. With the Year 2000 effort, scope adjustment is not an option, schedule adjustment is out of the question, and that leaves only one option. Effective project management practices must effectively manage resources.

It is imperative that industry and Government work together and use the best tools and experiences of both to accomplish the Year 2000 compliance modifications. We must learn from each other and remain focused on the final results needed. I believe that industry, while working to make its own systems compliant, can contribute greatly to the Government's solution by providing our skilled resources and considerable experiences, whether through outsourcing, current contract vehicles, by providing tools and methodologies, or through providing consulting services.

The time to address the issue is now. We cannot afford to wait any longer. The Year 2000 will occur on time, and we must be ready. Furthermore, industry eagerly awaits the authorization and appropriations legislative taskings to get underway. We are in a race against time in preparing for the Year 2000.

Thank you for inviting me to testify during today's hearing. I would be pleased to take any questions you might have.

Chairwoman MORELLA. Thank you, Mr. Ingram. And, the record will have within it the entirety of the statement that you have presented.

All right. Ms. McDuffie.

STATEMENT OF BARBARA L. McDUFFIE, IBM PROGRAM DIRECTOR OF SOLUTION PROVIDER PROGRAMS, POUGHKEEPSIE, NEW YORK

Ms. McDUFFIE. Thank you, Madam Chairwoman. I am Barbara McDuffie. I am the Program Director of Solution Provider Programs.

And, you probably don't know what that means. But, what I do for a living everyday is work with third party application providers to ensure that their programs work with IBM systems.

I want to thank you for allowing me to comment about this industry-wide Year 2000 challenge. I will confine my remarks to summarizing what IBM is doing in helping our customers solve this challenge.

IBM is proud to be taking a leadership role here in an issue which spans the entire computing industry. We have already taken a series of important steps to help our customers.

Fortunately, many companies and public organizations have a sense of urgency about Year 2000 compliance and are taking strong and immediate actions. And, I think you saw that with the testimony from SSA.

However, from IBM's perspective, many companies, institutions and organizations need to move more aggressively to make these necessary changes. If we were to use a marathon as an example, I think we can say that many businesses haven't even completed the first mile.

The most important thing that we, in IBM, are doing today is promoting the awareness among our customers through user conferences, teleconferences, briefings, consultant and press interviews. And, of course, we are using our IBM home page on the Internet, since that seems to be the magic that everyone believes for education today.

We want our customers to know what the Year 2000 challenge is. Why is this important? What must they do to become Year 2000 compliant? And, why must work start now rather than later?

A Year 2000 Customer Council, made up of a variety of IBM customers, have given us important recommendations on developing a planning and implementation guide. It is important to understand that this Year 2000 challenge has a smaller impact upon computer hardware and operating systems as compared to software applications.

For example, IBM mainframes and our mid-range computers that have been sold within the last several years are Year 2000 compliant. Now, what this means is that when the date changes at Year 2000, there will be no impact upon the timing mechanisms in those computers.

However, for older hardware, such as the 20 year old IBM non-ESA Capable System 370, it's not practical or effective to fix it to accommodate the four digit dates. Personal computers will vary—and I know probably many of you have experience with personal computers—as to how they will handle that change. But, it's relatively easy to set their internal clock and calendar so that they accept the Year 2000 date.

The most serious impact is with the software applications. And, you've heard that already here today.

Much of it is coded years ago, even by programmers in companies and organizations in both private and public sector. Almost all of these applications, data bases and other non-operating system software used two digits to represent the year.

IBM is among these companies, because we have a large number of internally written applications, many of them dating back many years that run our business everyday. We have evaluated these key applications and we are fast at work recoding and altering them to make them Year 2000 compliant by the end of next year.

It is part of a major application re-engineering effort inside IBM to make us more efficient as well as to deliver products and services much more quickly.

Actual technical fixes to the application software code may be numerous, but they are relatively straightforward. The real issue is for private and public sector and individual computer users to be aware of this problem, realize they must not wait to get started, prioritize—that's key—figure out which applications are key to your mission critical business and begin making the necessary coding changes.

Last fall, IBM announced its commitment to provide customers with a set of services, tools and support for Year 2000 transition. First, we want to make sure that our own hardware and current releases of software that we sell to customers are Year 2000 compliant.

And, we've made it our intention to have that done by year-end this year, 1996. And, we are on track to make that happen.

Secondly, we have started making available to anyone without charge a 200 page planning and implementation guide that offers step-by-step guidance on how to address this problem. This is available on the Internet.

It also lists IBM products and many vendor products that are Year 2000 compliant. I have also submitted this as additional written testimony.

Chairwoman MORELLA. Without objection, it will be included.

Ms. MCDUFFIE. Okay. We have given away thousands of copies to customers with more than 6,000 additional copies already downloaded from IBM's Internet web site. And, we continue to update this reference periodically.

Thirdly, we also have a set of available tools if customers want to buy those to assist them in updating their application programs.

And, finally, for those customers who seek additional help, as you mentioned Mr. Tanner earlier, IBM also announced last fall a set of fee-based services to help companies that want extra help in making their application software and data Year 2000 compliant.

Based on our discussions with customers, we believe more and more of them are becoming aware of the Year 2000 challenge. A growing number have begun to take steps to assess the magnitude of what applications they need to alter and how much work is involved.

Clearly, they are motivated to take explicit action. But, we believe customers are not—many customers are not as far along.

Thus, the news is mixed here. There is a growing awareness, more software tools are available and help is available.

But, the time for action is now. By acting now, businesses can make the changes to implement the necessary coding updates to continue their business operations.

Time is growing short. I want to thank you, Chair Morella, for holding this particular hearing, because I think it's important that we spread the visibility and the awareness of this issue.

And, I will be happy to take any questions from the panel.

[The prepared statement and attachments of Ms. McDuffie follow:]

TESTIMONY OF BARBARA L. McDUFFIE

IBM PROGRAM DIRECTOR OF SOLUTION PROVIDER PROGRAMS

HOUSE COMMITTEE ON SCIENCE

TECHNOLOGY SUBCOMMITTEE

MAY 14, 1996

Thank you, Madam Chair:

I am Barbara McDuffie, program director of solution provider programs for IBM. I want to thank you for allowing me to comment about the industry-wide Year/2000 challenge.

I will confine my remarks to summarizing what IBM is doing in helping its customers solve this challenge. IBM is proud to be taking a leadership role in an issue which spans the entire computing industry.

We have taken a series of important steps to help our customers. Fortunately, many companies and public organizations have a sense of urgency about becoming Year/2000 compliant and are taking strong and immediate actions.

However, from IBM's perspective, many companies, institutions and organizations need to move more aggressively to make the necessary changes. If we use a marathon as an example we can safely say many businesses haven't completed the first mile.

The most important thing IBM is doing is promoting awareness among our customers through user conferences, teleconferences, briefings, consultant and press interviews and of course the IBM home page on the Internet. We want our customers to know what the Year/2000 challenge is—why it is important—what they must do to become Year/2000 compliant and why work must start now rather than later. A Year/2000 Customer Council made up of a variety of IBM customers has given us important recommendations on developing a planning and implementation guide.

It is important to understand that the Year/2000 challenge has a smaller impact upon computer hardware and operating systems compared to software applications.

For example, IBM mainframes and mid-range computers sold within the last several years are Year/2000 compliant. What this means is that when the date changes to 2000 there will be no impact upon the timing mechanisms in these computers.

However, for older hardware, such as the 20-year-old IBM Series 370, it is not practical or cost effective to fix it to accommodate four-digit dates.

Personal computers will vary in how they handle the date change, but it is relatively easy to set their internal clock and calendar so they accept the Year 2000.

The most serious impact is with the software applications—much of it coded years or even decades ago by programmers in companies and organizations in both the private and public sector. Almost all of these applications, databases and other non-operating system software used two digits to represent the year.

IBM is among these companies because we have a large number of internally written applications—many of them dating back years. We have evaluated these key applications and are fast at work recoding and altering them to make them Year/2000 compliant by the end of next year.

Actual technical fixes to the application software code may be numerous, but are relatively straightforward. The real issue is for the private and public sector and individual computer users to be aware of the problem—realize they must not wait to get started—prioritize what critical applications must be fixed first—and begin making necessary coding changes.

Last fall IBM announced its commitment to provide customers with a set of services, tools and support for the Year 2000 transition.

First, we want to make sure the most recent version or release of IBM's software products that we sell are Year 2000 compliant by year-end 1996. We are on track for that.

Second, we started making available to anyone without charge a 200-page planning and implementation guide that offers step by step guidance on how to address the problem. It also lists IBM products that are Year/2000 compliant.

We have given away thousands of copies to customers with more than 6,000 additional copies already downloaded from IBM's Internet web site. We continue to update this reference work periodically.

Third, for those customers seeking additional help, IBM also announced last fall a set of fee-based services to help companies that want extra help in making their application software and data Year/2000 compliant.

Based upon discussions with our customers we believe that more and more of them are aware of the Year 2000 challenge. A growing number have begun taking initial steps to assess the magnitude of what applications they need to alter and how much work is involved. Clearly, they are motivated to take explicit action.

But we believe many customers are not as far along. Thus, the news is mixed. There is a growing awareness; mare software tools are available; and help is available.

But the time for action is NOW. By acting now businesses and organizations will be in a better position to implement the necessary software coding changes in their application software and continue their operations.

Time is growing short.

I'll be happy to answer any questions you may have.

Frequently Asked Questions and Answers on Year 2000 Challenge

Q1. What is the Year 2000 Challenge and how did it happen?

A1. The scope of the Year 2000 Challenge spans the entire Information Technology industry. The phenomenon exists because for decades it has been common practice to use two digits instead of four when writing dates. This carried over when writing computer programs, especially when it comes to minimizing expensive memory space and data entry time. However common this practice, it causes computer software performing arithmetic operations, comparisons or sorting of data fields to yield incorrect results when working with years beyond 1999.

Q2. Who does this impact?

A2. It is a significant challenge across the IT industry -- for any company, social or government agency, institution or individual using computers to accomplish a task. Any system or program, including desktop software, could be affected if two digits are used for year representation.

Q3. Can't a user just switch from using two digits to four?

A3. The process of making the change is fairly straightforward, but time consuming. Users must first determine whether the data that represents "year" is stored as two digits and then find all the applications that use this data. If only two digits are used, the file format must be changed to four digits. Every application program that stores or references this data must also be changed. Finding all the programs that reference this data and coordinating the change are what takes time.

Q4. What actually happens if the Year 2000 issue isn't corrected?

A4. Any computer calculation that involves a date -- such as a consumer credit card transaction -- a payroll billing -- an electric company statement -- a mortgage calculation and so forth could yield incorrect answers.

Q5. Why haven't we heard about this before now?

A5. Many companies, organizations and individuals are not aware of the Year 2000 Challenge or the need to start preparing for it as quickly as possible. IBM believes IT vendors need to make their customers aware of the situation and help them become Year 2000 Ready.

Q6. Why did this two-digit practice continue for so long?

A6. For decades frugality as been the rule for programmers trying to save storage space by using only two digits to represent a year when writing or executing an application. Even when memory became relatively inexpensive the problem was never viewed as critical. Also, once established it was difficult to initiate a four-digit format because it would mean changing all existing software. Spending money on a "software maintenance" issue may not have been given high priority.

Q7. Is this a hardware or system software problem or both?

A7. It is primarily a software application problem.

Q8. What specifically do you recommend computer users do?

A8. Computer users need to update applications and data fields that do not handle century markers or dates beyond 1999. Specifically they should:

- Determine the magnitude of the problem facing them by assessing their entire portfolio of system and application software source code, including any shrink-wrapped, off-the-shelf applications, to determine what needs to be updated and made Year 2000 Ready.

- Decide the best way to make the updates -- most likely on an individual, program by program basis.

- Implement the updates to the source code; test to make sure it handles both 199X and 2XXX data correctly; and establish a procedure to ensure the source code can't be inadvertently changed back to a two-digit format.

Q9. Why the rush -- why can't customers fix their problems and become Year 2000 Ready in 1998 or 1999?

A9. Many customers may run out of time and not be able to alter their application portfolio if they wait. Also, the Year 2000 problem is already beginning to surface for some customers and will occur more frequently as we approach the year 2000.

Q10. You've mentioned software a lot, what needs to be done to a computer user's hardware?

A10. Computer users should review their users manual or if necessary contact their vendor or sales representative to determine whether the internal timing mechanism in their computer hardware can handle the change of century.

For example, the hardware timers on IBM S/390*, AS/400*, RISC/6000* and PowerPC* machines are not sensitive to the change of century. The same will be true of all IBM Personal Systems* and IBM PC Server* models introduced after January 1, 1996. For current and older model IBM Personal Systems and PC Servers the requirements vary. Some may automatically make the change while some may require a utility instruction to be executed or a typed command entered to update the century.

Q11. Does the Year 2000 Challenge apply equally to the desktop world?

A11. Yes.

Q12. Are there estimates on how difficult, how long and how much it will cost a company or individual computer user to make the transition and become Year 2000 Ready?

A12. Making applications and system software Year 2000 Ready is a type of redevelopment project, the scope of which depends upon the size and amount of software being used. Gartner Group consultants have estimated that a typical mid-size company could spend as much as $3 million to $4 million in personnel and computer resources to make the changes. They add that large companies or organizations could spend ten times that or more. Desktop and Small Office/Home Office computer users will have to contact their vendors and suppliers to see what desktop software is Year 2000 Ready.

Q13. What products, services or information is IBM making available to help customers become Year 2000 ready?

A13. IBM is taking a leadership role in assisting customers and the IT industry in the conversion effort and becoming Year 2000 Ready. Specifically:

First, IBM has developed a reference guide called "Year 2000 and 2 Digit Dates: A Guide for Planning and Implementation" and is offering at no charge to anyone, by down loading it from the Internet. An extensive bibliography lists many of the most prominent publications and trade press articles concerning the Year 2000 Challenge. It is available on the World Wide Web through the IBM Software Group Home Page at http://www.software.ibm.com.

This reference guide will provide generic and IBM-specific information to help customers:

- Understand the cause and scope of the Year 2000 issue;

- Identify applications with years represented with two-digits.

- Plan for migrating to a Year 2000 environment;

- Decide best reformatting and testing techniques;

- Select and use appropriate IBM and Solution Developer tools

Second, IBM intends that, as of year-end 1996, the versions/releases of current IBM software products will support the Year 2000 and beyond.

Third, IBM is highlighting new and existing software tools to help customers with the transition. These tools and compilers are platform-specific and target the host application development environments. They support MVS*, OS/400*, AIX*, OS/2*, VSE* and VM* customers. A complete set of the tools, compilers and product listings is included in the IBM Year 2000 reference guide.

Fourth -- for customers requiring more help IBM is offering, through its Integrated System Solutions Corporation, a comprehensive set of solutions and services to help customers assess, plan and implement the steps necessary to become Year 2000 ready. The offerings will help reduce the cost and complexity of the transition.

* Indicates trademark or registered trademark of International Business Machines Corporation

News Release
IBM.

International Business Machines Corporation
IBM United States
1133 Westchester Avenue
White Plains, New York 10604

Immediate Release

Marta Decker
IBM Media Relations
914/892-7358

Jordan Chanofsky
TSI for IBM
212/696-2000

Year 2000 Efforts Under Way

IBM Readies Customers, Products and Services for Year 2000 Transition

Armonk, New York, October 30, 1995 ... Recognizing that the turn of the century poses a significant challenge for the Information Technology industry, IBM today announced it will provide customers with a comprehensive set of services, tools and support for their Year 2000 transitions.

For more than four decades, industry and businesses have written many of their computer programs and databases with dates represented by only two digit years (e.g., 95 versus 1995). However popular this method was, and is, customers' system and application programs may yield incorrect results when the millennium advances, and the date approaches "2000."

This means that customers whose businesses typically rely on applications which make forecasts, projections, comparisons, or arithmetic operations are encouraged to complete their preparations for Year 2000 date changes now.

- more -

The difficulty for many businesses comes in assessing what applications have date-sensitive programs; how many need to be altered; what it takes to actually make the required changes to source code and data files; and finally, running tests to ensure that all is operating properly. IBM's Year 2000 services, tools, and support will assist customers with this process.

"If customers are to be successful in tackling the Year 2000 issue, they need to focus on specific date-change methodologies, processes -- and overall project management," said John Phelps, Gartner Group. "Year 2000 projects need to be expedited by customers so that they can accurately determine their application programs' exposures and can begin corrective measures immediately."

"The problem is large; it's complex, and the IT industry has the skills and resources to take care of it -- providing we give ourselves the time to solve it," said Peter de Jager, Year 2000 consultant and speaker. "IBM is right to encourage and advise businesses, and vendors who support that business, to address this issue today."

Because of IBM's commitment to protect its customers' investments -- and its obvious interest in the long-term viability of the computer industry -- the company has spent considerable time researching, testing and analyzing the Year 2000 issue and possible solutions.

"With today's announcement, IBM is sharing what we have learned about the Year 2000 with our customers, and all computer

- more -

users, to help them make date transitions as smooth as possible," said Carla Gude, director of System Software Structure, IBM. "No matter how old or new their software is, customers and industry vendors will never know how much work is ahead of them -- unless they focus now."

Information, Services, Tools and Support

To assist customers in timely Year 2000 transitions, IBM has assembled a variety of information, services, tools and support.

The following Year 2000 offerings are being announced by IBM today:

Year 2000 Customer Guidance Paper

IBM is making available to everyone a comprehensive Year 2000 resource guide, at no charge. The guide explains Year 2000 issues and helps users, vendors and customers successfully plan for -- and implement Year 2000 transitions. The 180-page document, entitled "The Year 2000 and 2-Digit Dates: A Guide for Planning and Implementation," is available on the World Wide Web through the IBM Software Home Page, at http://www.software.ibm.com. Customers can also obtain the guide from their IBM marketing representatives.

This no-charge resource is a compilation of IBM's Year 2000 findings, recommended approaches and product listings. Also included in the guidance paper is a bibliography of other Year 2000 publications available throughout the industry.

TRANSFORMATION 2000* Services

In addition to the Customer Guidance Paper, IBM is making available to customers a comprehensive set of fee-based services

- more -

to help companies develop Year 2000-ready solutions for their applications, system software and hardware.

TRANSFORMATION 2000 services, delivered by Integrated Systems Solutions Corporation, an IBM subsidiary, are available to IBM and non-IBM clients operating in both centralized and distributed computing environments. These new services seek to balance customers' Year 2000 investment activities -- with their current and planned strategic business initiatives.

TRANSFORMATION 2000 solutions make date-field transitions easier by bringing together proven techniques and state-of-the-art technologies to help reduce cost, redundancy and complexity for the customer.

Year 2000-Ready Software

By year-end 1996, IBM intends to have the most recent versions and releases of current IBM software products supporting the Year 2000 and beyond.

In order to assist customers with planning for the analysis, updating and testing of user and vendor applications and data, IBM provides a table in the Customer Guidance Paper which lists many widely-used IBM products, and spells out the level or levels that will be 2000-ready. Many of these products are available now, and the others are being shipped between now and year-end 1996.

Year 2000 Tools

IBM is also highlighting new and existing software tools to assist customers with their Year 2000 transitions. These tools and compilers are platform-specific and target the host application development environment.

They support MVS*, OS/400*, AIX*, OS/2*, VSE* and VM* customers. A significant set of the tools, compilers and product listings is included in the Year 2000 Customer Guidance Paper.

Year 2000 Hardware Support

The hardware timers on IBM S/390*, AS/400*, and RISC System/6000* servers and Personal Systems*, using PowerPC* technology, are not effected by Year 2000 date changes.

IBM Personal Systems and IBM PC Servers introduced in 1996 will handle the century rollover automatically. Some current and earlier IBM PCs will automatically update the century; others may need to enter a simple command or use a special utility. These systems need to be tested because there are different BIOS -- or basic input/output systems, handling the timing routine. Diagnostic guidance and tools will also be available to help users understand what to do for their individual IBM systems.

Date functions in some IBM network devices are currently being updated as well.

In regard to IBM and other vendors' personal computers and systems, a variety of implementations exist and behavior is system-dependent. For non-IBM PCs, some of the same procedures

- more -

used for IBM systems may also be useful to determine if changes to customers' systems are required. In addition, customers are encouraged to contact individual vendors regarding their non-IBM products and related compliance questions.

Year 2000 Information at Your Fingertips

Information on IBM and ISSC's Year 2000 services, tools and support can be obtained on the Internet via IBM's Software Home Page on the Worldwide Web. The Software Home Page is located at http://www.software.ibm.com.

The Customer Guidance Paper and its "White Paper" summary are also available in several formats via the IBM Software Home Page.

The IBM FAX Information Service allows you to receive facsimiles of this, and other IBM product press releases. Dial 1 800 IBM-4FAX and enter "99" at the voice menu. From outside of North America, facsimiles may be obtained by calling 1-415-855-4329.

# # #

* Indicates trademark or registered trademark of International Business Machines Corporation.

YEAR 2000: A PERSPECTIVE

If your computer system or its applications use two digits to represent the year, and many do, the change to dates of 1999, 2000 and beyond may skew the accuracy of the data created by every application from word processors to databases.

If it is not addressed quickly, this date change could affect calculations, comparisons and data sorting in applications from the desktop on up to the largest server. The price tag for potential errors could be high and could have a major impact on commercial, industrial, educational and governmental operations of all types.

This is a challenge for a world economy that depends on information technology. The problem can exist in any level of hardware or software, in the microcode, in new and old applications, in files and databases and on any computing platform.

IBM has taken a lead in working with others in the Information Technology industry to develop methodologies and solutions to solve the problem. IBM has just published a detailed guide for information system managers, programmers, consultants, and computer users in general who will be dealing with Year 2000 conversions. Titled "The Year 2000 and 2-Digit Dates: A Guide fór Planning and Implementation," it is available at no charge on the Web through the IBM Software Group Home Page at http://www.software.ibm.com.

In clear, simple language, the guide explains the parameters of a problem that is easy to grasp, but not as easy to solve. Many computer operating systems and applications use a standard two-digit format, MM/DD/YY, to generate a date. A computer would write January 1, 1996 as 01/01/96. January 1, 1999 would be 01/01/99. When the year rolls over to 2000, computers that use the two-digit format might write 01/01/00. And therein lies the problem.

Many programs calculate the length of time between dates by subtracting two-digit years from each other. For example, a bank computer does this simple calculation when it figures the principal and interest on a 30-year mortgage. There is no problem if the mortgage was issued in 1965 and paid in full in 1995. But if the mortgage is issued today and the application uses the two-digit date format, it may well read the year as "00" and be unable to do the calculation. The application may assume the year is 1900 and return an error message.

The same problem occurs with invoices, payrolls, credit card transactions, bill payments, inventory systems, auto loans, databases that sort by year, and many other computer operations upon which daily life depends.

Why wasn't it fixed before now? Often date fields can't easily be expanded to include more than two numbers. Nor can computers be instructed to globally insert the digits "one" and "nine" into years, because some dates may refer to a different century. One solution is to painstakingly find every point in a system that uses the date to trigger a calculation or a routine and rewrite the source code. All applications must be changed in a coordinated fashion and tested to make sure they handle dates in both centuries.

Why didn't the Information Technology industry foresee from the beginning that two-digit dates would be a problem and use four digits to represent the year?

In the 1960s and 1970s, when many computer programs were created, programmers had incentives to abbreviate dates and save data entry time. Computer memory and storage were costly and in short supply. Saving several characters in every database with millions of records was worth it. Early programming languages and standards didn't offer application program interfaces (APIs) for four-digit years. And even if they were aware of the pitfall, programmers may have assumed that the applications they were writing would be scrapped long before they could cause problems. Remarkably, many of those early programs are still in use, often as the core of company information systems, and that data is still in the same format.

A few industries have been working on the two-digit date problem since 1970. That is when bank application and insurance programs first began to have difficulties with amortization and interest table calculations.

Why have many companies waited to grapple with the date problem? The task of fixing it is daunting. Organizations may need to analyze, change, and test millions of lines of code, many hundreds of applications and many thousands of routines and sub-routines, all done in a coordinated effort. It could also be very expensive. The Gartner Group, which consults on information technology, estimates that a medium-sized company with 8,000 programs will spend between $3.6 and

$4.2 million to repair "date-challenged" software.

We have become so dependent on technology that no one--individual or large corporation--can afford the consequences of not making the investment. The authors of the IBM planning guide stress that solving the two-digit date problem is not technically difficult. The project may be large, but like most tasks, it can be managed with careful planning. The guide offers clear instructions on how to plan and test systems and how to reformat them. IBM and other developers also have a wide array of solutions, tools, services and support.

By year-end 1996, IBM intends to have the most recent versions of its software products able to work with dates from the next century. Hardware timers for many IBM platforms, including S/390*, AS/400*, RISC System/6000*, and IBM Personal Systems* using PowerPC* technology are not affected by the end-of-century problem. All of IBM's personal systems and IBM PC Servers introduced after January 1, 1996, will automatically update the century. Earlier IBM PC Servers* and IBM Personal Systems need to be tested because there are different BIOS, or basic input/output system, handling the timing routine. For some, users may need to enter a command or use a special utility.

The IBM guide has simple steps to test whether a personal computer will update to the new century. The IBM guide also shows how to test the personal computer's operating system and time-sensitive programs.

IBM also has a range of tools to help its customers make the transition to the Year 2000.

There is an extensive set of tools for the popular COBOL programming language used in IBM's large-scale and mid-size platforms, some aimed specifically at host application re-development. Other tools address more languages on these platforms.

For companies seeking additional services or help, IBM's Integrated System Solutions Corporation has a broad range of offerings. Called Transformation 2000 Solutions, this approach can help enterprises solve two-digit date problems and, at the same time, help to make their information systems more efficient.

Companies often need to analyze their entire technology infrastructure in detail to get ready for the Year 2000. Transformation 2000 offerings include assessing a company's application portfolio, developing a strategy, planning and implementing it. IBM also will test application modifications after the changes are made.

The Year 2000 is fewer than a thousand working days away. Now is the time to update programs and systems to make the change. Any later could be too late.

Third Edition

GC28-1251-02

The Year 2000 and 2-Digit Dates: A Guide for Planning and Implementation

Note

Before using this information and the product it supports, be sure to read the general information under "Notices" on page xv

Third Edition, May 1996

This is a major revision of, and obsoletes, GC28-1251-01

Order publications through your IBM representative or the IBM branch office serving your locality Publications are not stocked at the address below

IBM welcomes your comments A form for readers' comments may be provided at the back of this publication, or you may address your comments to the following address

International Business Machines Corporation
Department 55JA, Mail Station P384
522 South Road
Poughkeepsie, NY 12601-5400
United States of America

FAX (United States & Canada) 1+914+432-9405
FAX (Other Countries)
Your International Access Code +1+914+432-9405

IBMLink (United States customers only) KGNVMC(MHVRCFS)
IBM Mail Exchange USIB6TC9 at IBMMAIL
Internet e-mail mhvrcfs@vnet ibm.com
World Wide Web http //www s390 ibm com/

If you would like a reply, be sure to include your name, address, telephone number, or FAX number

Make sure to include the following in your comment or note

- Title and order number of this book
- Page number or topic related to your comment

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you

Limited rights to copy the present work are hereby granted by the copyright owner named below Accordingly, there is hereby granted the right to make a limited number of additional copies solely for the internal convenience of the recipient, no copies may otherwise be made In particular, no copies may be made, no derivative works may be created and no compilations of the subject work may be created for purposes of republication, for redistribution, for sale, for rental, for lease or for any profit motivated activity whatsoever including the use of this work in support of or in conjunction with any service or service offering

Note to U S Government Users — Documentation related to restricted rights — Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract with IBM Corp

©Copyright International Business Machines Corporation 1995, 1996

Contents

About This Book ix
Who Should Use This Book ix
How to Use This Book
  Of General Interest to Everyone x
  Executive and Senior-Level Management x
  IS Managers x
  System Programmers x
  Application Programmers xi
Executive Summary xii
  Does this Really Mean Me? xii
  But I've Been Told xii
  What's My Role and What Can I Expect? xiii
  What Are IBM and the Solution Developers Doing to Assist Me? xiii
  So, What's the Bottom Line? xiv

Notices xv
Trademarks xv
  IBM Trademarks xv
  Lotus Trademarks xvi
  Non-IBM Trademarks xvii

Summary of Changes xix

Chapter 1. The Year 2000 - A Transition 1-1
Year2000 Exposure Classification 1-2
Scope of Year2000 Transition 1-3

Chapter 2. Planning to Resolve Your Year2000 Exposures 2-1
Planning Considerations 2-2
Inventory Your Software Portfolio 2-6

Chapter 3. Identifying 2-Digit-Year Exposures 3-1
Locating References 3-1
  Tracing References Back to Their Source 3-2
Determining the Impact of 2-Digit-Year Data Fields 3-3
Investigating How Other Software Entities Use the Data 3-3
  Data Sharing 3-4

Chapter 4. Reformatting Year-Date Notation 4-1
Solutions and Techniques 4-1
  Solution #1 Conversion to Full 4-Digit-Year Format 4-1
  Solution #2 Windowing Techniques 4-3
  Solution #3. A 2-Digit Encoding/Compression Scheme 4-6
Using a Common Date/Time Service Routine 4-9
Considerations When Selecting Solutions 4-9
  Solution Applicability 4-10
  Bridge Programs Help Stage Format Conversions 4-10
  Other Programming Situations 4-11
Guidelines 4-12

Chapter 5. Testing Techniques for Year2000 Changes 5-1

© Copyright IBM Corp 1995, 1996

Structural Testing Techniques 5-1
  Operations Testing 5-1
  Stress Testing 5-1
  Recovery Testing 5-2
Functional Testing Techniques 5-2
  Requirements Testing 5-2
  Regression Testing 5-2
  Error Handling Testing 5-3
  Manual Support Testing 5-3
  Intersystem Testing 5-3
  Parallel Testing 5-3
How to Change Date and Time for Testing 5-4
Basic Testing Scenarios 5-6
  Basic Scenarios to Test Your PC System Clock 5-7

Chapter 6. Migration Consideration for Year2000 Transition 6-1
Plan for Migration 6-1
Perform Migration 6-3

Chapter 7. Tool Categories and Available Tools to Ease Year2000 Changes 7-1
Tool Characteristics 7-1
Tool Categories 7-2
  Impact Analysis 7-2
  Project Management 7-3
  Program Level Analysis 7-3
  Code Editing and Restructuring 7-4
  Code Generation 7-5
  Automate Testing 7-5
IBM Tools for MVS 7-7
  The IBM COBOL Family for MVS & VM 7-7
  The IBM PL/I Family for MVS & VM 7-18
  DFSORT 7-23
  COMUDAS (COMmon Uithoorn DAte Services) 7-26
IBM Tools for VM 7-29
  The IBM COBOL Family for MVS & VM 7-29
  The IBM PL/I Family for MVS & VM 7-29
  REXX/EXEC Migration Tool for VM/ESA 7-29
IBM Tools for VSE 7-31
  The IBM COBOL Family for VSE 7-31
  The IBM PL/I Family for VSE 7-36
IBM Tools for AS/400 7-39
  Using OPM RPG/400 Date Support 7-46
  Using System/36 Compatible Date Support 7-46
  The IBM COBOL Family for AS/400 7-47
  The IBM C Family for AS/400 7-47
  Integrated Language Environment for OS/400 7-48
  DB2/400 SQL 7-48
  OS/400 CL 7-54
  Application Dictionary Services/400 7-59
  Application Development Manager/400 7-59
IBM Tools for Personal Computers 7-61
  The IBM COBOL Family for the Workstations 7-61
  The IBM PL/I Family for the Workstations 7-67
Solution Developer Tools 7-71
  Solution Developer Contacts and Product Name 7-71

Chapter 8. IBM Consulting and Services 8-1
TRANSFORMATION 2000 IBM's Century Date Change Solutions 8-1
TRANSFORMATION 2000 Solutions 8-1
  Assessment and Strategy 8-1
  Detailed Analysis and Planning 8-1
  Implementation 8-2
  Year 2000 Clean Management 8-2
Summary 8-2

Appendix A. IBM Year2000-Ready Key Program Products and Hardware A-1
Exception Considerations and Program Products A-1
  Environmental Report Editing and Printing (EREP) A-1
MVS A-3
TPF A-8
VSE/ESA A-9
VM A-11
OS/400 A-13
AIX A-16
OS/2 A-20
Lotus Products A-22
Hardware A-24
  IBM Personal Computer (PCs) - Hardware Timer Setting A-24
  Desktop PC Systems A-24
  Commercial PC Desktop Systems A-30
  Mobile PC Systems A-33
  PC Servers A-35

Appendix B. Year2000-Ready Solution Developer Products B-1

Appendix C. Bibliography C-1
Non-IBM Publications C-1
  By Author C-1
  By Title C-5
  Electronic (Internet/World-Wide Web) Documentation C-8
  Audios and Videos C-11
IBM Publications C-12
  By Author C-12
  Standard Publications C-12

Appendix D. Glossary D-1

Index X-1

Year2000-Ready Solution Developer Product and Tool Authorization X-3

Figures

1-1 A Corporate Networked Computing Environment 1-4
4-1 Graphical Representation of the Sliding Window Technique 4-5
4-2 Example User-Defined Date/Year Conversion Table 4-7
7-1 Date Formats for Date Data Type 7-41
7-2 Date Values 7-41
7-3 Time formats for Time data type 7-42
7-4 Time Values 7-42
7-5 Solution Developer Tools 7-71
A-1. MVS Platform Products A-3
A-2 TPF Platform Products A-8
A-3 VSE Platform Products A-9
A-4. VM Platform Products A-11
A-5 OS/400 Platform Products A-13
A-6. AIX Platform Products A-16
A-7 OS/2 Platform Products A-20
A-8 Lotus Products A-22
A-9. Desktop PC Systems - Internal Clock Setting A-24
A-10 Commercial Desktop PC Systems - Internal Clock Setting A-31
A-11 Commercial Desktop PC Systems - Internal Clock Setting A-32
A-12 Mobil PC Systems - Internal Clock Setting A-33
A-13. PC Servers - Internal Clock Setting A-35
B-1 Year2000-Ready Solution Developer Products B-1

© Copyright IBM Corp. 1995, 1996

About This Book

This book provides information to help you:

- Understand the cause and scope of using dates represented by two-digit years
- Identify problems with programs using 2-digit data
- Plan for your installation's migration to a Year2000-ready environment
- Determine the best technique(s) for reformatting your year-date notation
- Determine the best technique(s) for testing your Year2000 changes
- Select and use IBM and Solution Developer tools appropriate for your needs

The official copy of the book is the on-line version. You can access this document through an anonymous ftp whose URL is:

`ftp://lscftp.kgn.ibm.com/pub/year2000/y2kpaper.`**`format_type`**

where **format_type** can be one of the following print formats:

Print Format	***format_type***
ASCII text	txt
Browseable BookManager	book
List3820	list3820
PostScript	ps
UNIX compressed	ps z
Tersed listps	tersps
Tersed list3820	ters3820

Refer to "Electronic (Internet/World-Wide Web) Documentation" on page C-8 for further information and related IBM on-line documentation.

Who Should Use This Book

The Year 2000 and 2-Digit Dates: A Guide for Planning and Implementation is directed, but not limited to, the following audience:

- Executives requiring an overview of the Year2000 challenge
- Installation managers and system and application programmers needing information to plan, install, modify and test their applications and systems to become Year2000-ready
- Users of IBM and non-IBM products and services
- Solution Developers that have software products on IBM and non-IBM platforms
- Year2000 focal points and marketing representatives that provide a source of information for both internal and external queries related to the Year2000 phenomenon
- Independent consulting groups
- Programmers who generate and maintain 'in-house' application code

© Copyright IBM Corp. 1995, 1996

How to Use This Book

The following categories are designed to help you determine which chapters are most suited to your specific needs.

Of General Interest to Everyone

To help your overall understanding of the challenge, you should

- Read Chapter 1, "The Year 2000 - A Transition" on page 1-1
- Scan Appendix C, "Bibliography" on page C-1 for additional readings related to topics that range from an overview of the Year2000 challenge to management issues to in-depth technical papers and discussions through both written and electronic (Internet) media

Executive and Senior-Level Management

You should read "Executive Summary" on page xii to learn about the challenges facing the Information Technology (IT) community and the challenge facing **your** information system (IS) staff

IS Managers

You will find it valuable to read the entire guide to learn more about which actions you need to take in support of your system and application programming staff.

Minimally, however, you should read the following:

- To obtain a high-level overview of the Year2000 challenge, read
  - "Executive Summary" on page xii
  - Chapter 1, "The Year 2000 - A Transition" on page 1-1
- For project planning information and recommendations, read Chapter 2, "Planning to Resolve Your Year2000 Exposures" on page 2-1
- For migration information and recommendations, read Chapter 6, "Migration Consideration for Year2000 Transition" on page 6-1.
- For a list of IBM and Solution Developer tools available to assist in identifying, coding, and testing Year2000 changes, read Chapter 7, "Tool Categories and Available Tools to Ease Year2000 Changes" on page 7-1.

System Programmers

- For an overview, you should read Chapter 1, "The Year 2000 - A Transition" on page 1-1.
- To determine the best technique(s) for reformatting year-date notation, you should read Chapter 4, "Reformatting Year-Date Notation" on page 4-1.
- For migration information and recommendations, read Chapter 6, "Migration Consideration for Year2000 Transition" on page 6-1.
- To assist you in testing your Year2000 modifications, you should read:
  - Chapter 5, "Testing Techniques for Year2000 Changes" on page 5-1
  - Chapter 7, "Tool Categories and Available Tools to Ease Year2000 Changes" on page 7-1

Application Programmers

- For an overview, you should read Chapter 1, "The Year 2000 - A Transition" on page 1-1.
- To understand how to identify the potential exposures caused by using 2-digit-year representations of dates, read Chapter 3, "Identifying 2-Digit-Year Exposures" on page 3-1
- To determine the best technique(s) for reformatting year-date notation, you should read Chapter 4, "Reformatting Year-Date Notation" on page 4-1
- To assist in your testing of your Year2000 modifications, you should read
  - Chapter 5, "Testing Techniques for Year2000 Changes" on page 5-1
  - Chapter 7, "Tool Categories and Available Tools to Ease Year2000 Changes" on page 7-1.

Executive Summary

A phenomenon exists in the Information Technology (IT) industry because historically many computer programs make use of dates represented by only two digits (for example, 95 rather than 1995) However common this practice might be, it causes programs (both system and application) that perform arithmetic operations, comparisons or sorting of date fields to yield incorrect results when working with years outside the range of 1900-1999 IBM refers to this phenomenon as the **Year2000 Challenge**

Does this Really Mean Me?

The scope of the Year2000 challenge spans the entire IT industry A data mismatch can exist in any level of hardware or software from microcode to application programs, in files and databases, and is present on **all** platforms (IBM and non-IBM). In recent years, the IT trade press has given ever greater attention to this phenomenon with increasingly ominous article titles

However dramatic all this may sound, consider the following scenarios to help put the phenomenon and its business ramifications into perspective Imagine if in the first quarter of the year 2000 your company cannot process its 1999 end-of-year billing or end-of-year payroll properly, your corporate credit card holders are refused most transactions because their accounts appear delinquent, your 1999 year-end profit data cannot be calculated properly; and your utility companies cut off their services due to your apparent late bill payments Similarly, your household and personal financial situation could encounter a similar dilemma if your creditors do not also strive to meet this challenge

Although referred to as the Year2000 challenge and often as the 'Year2000 Problem', this is really a 2-digit-year problem. Your IS organization needs to plan for, and address, the date changes well in advance of 1 January 2000 This is not only a future challenge, it has existed in the banking industry since as early as 1970 when application programs encountered problems with amortization and interest table calculations for the standard 30-year mortgage. Consider also, the standard 5-year automobile loan, a 15-year mortgage, a long-term insurance policy, a data base that retains birth dates (which includes an ever increasing set of dates over 100 years). Even before 1 January 2000, your computing environment might not function as expected because your systems and/or application programs will not process dates later than 31 December 1999 properly

But I've Been Told...

There are many misconceptions associated with the cause of the Year2000 challenge These include

- This is a problem that occurs only after 1999.

  As noted above, expect to experience problems well in advance of the year 2000 Application programs manipulating such data are likely to produce incorrect results today

- This is a hardware problem rather than the problem for your IS organization

  To the contrary, this problem comes mostly from application programs and data using two digits for year representations These programs have been purchased from Solution Developers, written in-house by your system and application programmers, leased from various software vendors, or shared among the IS community. Frugality was the rule when such programs were first written and every byte of saved storage was important, and human nature seems to dictate that writing and keyboard entry of two digits is more practical than four digits. Saving even a small amount of space in a data base was more important in the 1960's and 1970's than it is today, and most programmers never expected such code to still be in use 20 or 30 years later. This short-sighted programming practice has only recently surfaced as a problem, and only a rare program provided for four digits for the Year2000

- This is a problem that only occurs on mainframe systems and/or legacy applications

  Any system or program (large or small) can be affected if it uses two digits for year representation. In fact, such problems have been encountered in recently developed desktop software as well.

What's My Role and What Can I Expect?

For many institutions, you can expect the initial size of this Year2000 work effort to appear overwhelming and the overall project to appear enormous, as it should. This is not a trivial project. Expect the need for your sponsorship to dedicate personnel, dedicate computer resources, approve the requisition of potential Solution Developer tools and/or service support, and a budget to fund it all.

The solutions provided here are not particularly difficult to implement, and the technical solutions are not complex when viewed on a program-by-program basis. It is the overall project size and the program/data inter-relationships that introduces the true project complexity. Expect the need to appoint a cross-department and cross-divisional focal point to manage it all

What Are IBM and the Solution Developers Doing to Assist Me?

The project is manageable within the available time-frame and support is currently available from both IBM and many Solution Developers. IBM is committed to address the Year2000 Challenge within its own product offerings, to deliver Year2000-ready software, and to assist you in your efforts.

The IBM document (*The Year 2000 and 2-Digit Dates: A Guide for Planning and Implementation*), for which this summary is provided, is the result of an IBM multi-divisional effort to articulate the overall IBM support with respect to the impending Year2000 Transition. It is intended to provide the Information Systems (IS) community with planning, methodology, and guidance information needed to convert their application systems to correctly manipulate dates outside the range of 1900-1999

The Year 2000 and 2-Digit Dates: A Guide for Planning and Implementation is available through your IBM representative or in softcopy form through the Internet. That document provides your IS personnel both generic and IBM-specific information on:

- How to plan a Year2000 transition
- How to detect and remove Year2000 exposures
- How to test and migrate to a revised Year2000-ready system
- Where to find further information. (An extensive bibliography lists many of the most prominent publications and trade press articles concerning the Year2000 challenge.)

This document also provides information that addresses

- **Multiple Platform Solutions** (including IBM product-specific information for)
  - All IBM and non-IBM IS platforms
  - All IBM operating systems
  - IBM programming languages
- **Current IS Tools** (from IBM and non-IBM software vendors)

  Tools that are available today to help your IS personnel identify potential exposures in your applications, assist their efforts in the categorization of potential exposures, and assist the testing and transition to a Year2000-ready application environment
- **Consulting Services:** IBM consulting service offerings that provide skills and expertise for the understanding and/or the solution of your applications' Year2000 transition

Although this paper is designed mainly to provide in-depth information to your IS technical staff (that is, your applications and systems programmers), the support that it demands must come from you This will not be a small undertaking for your IS staff within any aspect of your IS environment If your business relies on computer processing, and that processing includes date manipulation, your business may be at risk This paper provides direction for your organization to begin a pro-active response to this challenge

So, What's the Bottom Line?

Your immediate attention to this effort, and a directive to your IS organization to plan for and solve your Year2000 challenge, is critical When you face your stockholders who will ask, "Now that you spent xxx dollars on this project, what do we have now that we didn't have in 1995?," simply respond "**We still have a viable business**, one that accepts billing, correctly computes interest and invoices, and calculates our stock dividends Not all our competition can make that same claim!"

Notices

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates.

Any reference to an IBM product, program, or service is not intended to state or imply that only IBM's product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any of IBM's intellectual property rights may be used instead of the IBM product, program, or service. Evaluation and verification of operation in conjunction with other products, programs, or services except those expressly designated by IBM, are the user's responsibility.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
500 Columbus Avenue
Thornwood, New York 10594
USA

Trademarks

IBM Trademarks

The following terms are trademarks of the IBM Corporation in the United States and/or other countries:

- ACF/VTAM
- AD/Cycle
- ADSTAR
- AFP
- AIX
- AIXwindows
- AIX/6000
- AnyNet
- APL2
- AS/400
- AT
- BookManager
- BookMaster
- C Set ++
- CallCoordinator/2
- CallPath
- CallPath SwitchServer/2
- CallPath/400
- CICS
- CICS/ESA
- CICSPlex
- CICS/MVS
- CICS/VSE
- COBOL/370
- COBOL/400
- CUA
- C/370
- C/400
- DATABASE 2
- DataHub
- DataPropagator
- DB2
- DB2/400
- DFSMS
- DFSMSdfp
- DFSMSdss
- DFSMSdhm
- DFSMS/MVS
- DFSMSrmm
- DFSMS/VM
- DFSORT
- DirectTalk/2
- DirectTalk/6000
- EOCF/2
- ESCON
- FASTService

© Copyright IBM Corp. 1995, 1996

- FFST
- FlowMark
- Facsimile Support/400
- GDDM
- geoGPG/6000
- HACMP/6000
- Hardware Configuration Definition
- IBM
- ImagePlus
- IMS
- IMS/ESA
- InfoCrafter
- InfoExplorer
- Integrated Language Environment
- ISPF
- ISSC
- LANDP
- Language Environment
- LoadLeveler
- MERVA
- MQSeries
- MVS/DFP
- MVS/ESA
- NetDoor
- NetView
- NTuneNCP
- OfficeVision
- OfficeVision/MVS
- OfficeVision/400
- OPC
- OpenEdition
- Operating System/400
- OS/2
- OS/400
- PowerPC
- PSF
- QMF
- RACF
- Resource Measurement Facility
- RISC System/6000
- RMF
- RMONitor
- RPG/4000
- RS/6000
- SMARTdata UTILITIES
- SOMobjects
- SP
- SQL/DS
- SwitchServer/2
- Sysplex Timer
- SystemView
- System/38
- S/390
- TRANSFORMATION 2000
- Trouble Ticket
- Ultimedia
- YEAR 2000
- VisualGen
- VisualLift
- VM/ESA
- VSE/ESA

Lotus Trademarks

The following terms are trademarks of the Lotus Development Corp in the United States and/or other countries

- Approach
- cc Mail
- Freelance Graphics
- Lotus
- Notes
- Organizer
- ScreenCAM
- SmartCenter
- Soft-Switch
- Word Pro
- 1-2-3

Non-IBM Trademarks

The following terms are trademarks of other companies as follows

Trademark	Company
ADAMS/400	Namtrig Incorporated
C/QUE	Systemware
Canada's National Newspaper	Globe and Mail
COBOL Analyst	SEEC, Inc
Change Action	Mazda Computer Corp
DateServer	Computer Software Corp
Globe and Mail	Globe and Mail
IntelliCONSOLE	IntelliWare Systems, Inc
IntelliRESOURCE	IntelliWare Systems, Inc
JHS	Systemware
MPS	Systemware
The OLR API	Data Base Architects, Inc
The OLR System	Data Base Architects, Inc
OnLine Help	Data Base Architects, Inc
OnLine Note pad	Data Base Architects, Inc
OnLine Reference	Data Base Architects, Inc
Portal 2000	DTS Software, Inc
RECY2000	Informission Group, Inc
REFINE Language Tools	Reasoning Systems
Challenge 2000	Micro Focus
Software Refinery	Reasoning Systems
SuperWylbur	SuperWylbur Systems, Inc
SystemVision	ADPAC Corp
TransCentury	TransCentury Data Systems
Version Merger	Princeton Softech
X/PTR	Systemware

Summary of Changes

Summary of Changes
for GC28-1251-02

This book contains information previously presented in *The Year 2000 and 2-Digit Dates A Guide for Planning and Implementation*, GC28-1251-01

The following summarizes the changes to that information

New Information

- Appendix B "Year2000-Ready Solution Developer Products" on page B-1 lists non-IBM, Year2000-ready products Solution Developers have requested IBM include in this document

Changed Information

- Appendix A, "IBM Year2000-Ready Key Program Products and Hardware" on page A-1 that lists those IBM products scheduled to be Year2000 ready by year-end 1996 has been expanded
- Appendix C, "Bibliography" on page C-1 has been updated to reflect current press releases, magazine articles, and currently available electronic documentation related to the Year2000 challenge

Summary of Changes
for GC28-1251-01

This book contains information previously presented in *The Year 2000 and 2-Digit Dates A Guide for Planning and Implementation*, GC28-1251-00

The following summarizes the changes to that information.

New Information

- An additional encoding scheme, the use of the `CYY` and `CCYY` is added to "Solution #3 A 2-Digit Encoding/Compression Scheme" on page 4-6 in Chapter 4, "Reformatting Year-Date Notation."
- DFSORT/MVS V1R13 will be enhanced to provide the ability to sort, merge, and transform 2-digit year data using a sliding window technique "DFSORT" on page 7-23 in Chapter 7, "Tool Categories and Available Tools to Ease Year2000 Changes" provides a detailed description of the new function and the enhanced control statements used to implement them
- Appendix C, "Bibliography" on page C-1 has been updated to reflect current press releases, magazine articles, and currently available electronic documentation related to the Year2000 challenge
- An authorization form, "Year2000-Ready Solution Developer Product and Tool Authorization" is provided in the back of this document for Solution Developers to request that IBM list their Solution Developer product and tool offering in future updates to this or related documents.

© Copyright IBM Corp 1995, 1996

Changed Information

- Appendix A, "IBM Year2000-Ready Key Program Products and Hardware" on page A-1 that lists those IBM products scheduled to be year2000 ready by year-end 1996 has been expanded
- Figure 7-5 on page 7-71 that lists Solution Developer products available to assist your year2000 resolution has been expanded

This book includes terminology, maintenance, and editorial changes Technical changes or additions to the text and illustrations are indicated by a vertical line to the left of the change

Chapter 1. The Year 2000 - A Transition

Although 1 January 2000 is still several years away, you need to plan for and address the 1999 to 2000 date change well in advance of that date. Even before we reach 1 January 2000, your computing environment might not work as expected because your systems and/or application programs will not process dates later than 31 December 1999 properly. However dramatic the following sounds, consider these scenarios to help put the problem into perspective. Imagine: all stock exchanges close down indefinitely because of invalid transactions, credit card companies refuse most transactions because cardholders appear delinquent on their payments; mortgage companies are besieged by angry borrowers who have just received delinquency notices in error and been charged extra interest, and utility companies cut off service to many customers due to apparent late bill payments.

The potential exposure mostly comes from, but is not limited to, the use of a 2-digit-year (YY) format, instead of a 4-digit (YYYY) format, for year representation within programs, files, databases, and processes. For example, the year 1996 is represented as '96', the year 1999 as '99', and so on. Thus, January 1, 2000 is represented as 01/01/00 (if using MMDDYY format in your data) and might be interpreted as January 1, 1900. This causes programs that perform arithmetic operations, comparisons, or sorting of date fields to yield incorrect results when manipulating year-date data of 2000 and beyond. The potential exposures that may be encountered, and their variations (listed in Section "Year2000 Exposure Classification" on page 1-2) have been referred to by IBM as the Year2000 challenge.

The scope of the Year2000 challenge spans the entire Information Technology industry. A data mismatch can exist in any level of hardware or software, from microcode to applications, in files and databases, and is present on **all** (IBM and non-IBM) Information System (IS) platforms.

Some general **misconceptions** of the Year2000 challenge include:

1. This is a problem that occurs only when/after the century rolls over.

   The challenges of handling the Year2000 changes may occur well before the year 2000 arrives. Typically, forecasting applications that deal with future dates will encounter problems well in advance of the year 2000. In fact, applications are already facing the problem; for example, the financial applications that deal with life insurance or bond policies that have expiration dates that go beyond the year 2000. In such an application, the programs need to handle the dates beyond the year 2000 when checking policy expiration.

2. This is a hardware clock problem that should be resolved by the computer vendors.

   To the contrary, this problem comes mostly from application programs and data using two digits for year representations even though the hardware clock and/or system timer services can provide a 4-digit-year format.

3. This is a problem that only occurs on mainframe systems and/or legacy applications.

   Any system or program can be affected if it uses only two digits for year representation: any file, database, log with 2-digit-year fields, and any data

© Copyright IBM Corp. 1995, 1996

entry, query, update and output processing that employs 2-digit-year fields. It can also affect programs, systems databases, and other functions and processing that receive input from these programs and data

Year2000 Exposure Classification

The Year2000 phenomenon is further compounded by the numerous variations used to express year and date notation in data and the mathematical calculations performed on those date notations These variations can be classified as the following exposure types

- Incorrect century[1]

  Problems can occur when the first two digits in a year are assumed to be 19 and

  - Ignored during data entry and update, or
  - Hard-coded for output

- Dates used as a special value

  Special values of the last two digits in a year might be used for special purpose, for example 99, 365/99, or 12/31/99 might be used to indicate "no expiration date" or 00 to indicate an "unknown year."

- Incorrect field format determination

  Some existing programs determine the date-time format (that is, MMDDYY, DDMMYY, YYMMDD) by testing an appropriate part of the date field For example checking if the first two characters of the date field are values within an acceptable month, day, or year range (such as 1-12, 1-31, or ≥32)

- Arithmetic calculation.

  The arithmetic calculations that operate on dates with 2-digit year representation might have potential exposures For example, a person with a birthday of 10 November 1961 will be considered to be −61 years old rather than 39 years old on 10 November 2000 if the years 1961 and 2000 are represented by 61 and 00, respectively Generally, a program that is not expecting a signed value, will ignore the sign, thus, in this example, the −61 might be interpreted as the absolute value 61 Not only is the value still incorrect, it is even less detectable than the incorrect −61 Further, if such incorrect data is then stored in a data base, it is considered a *data integrity* exposure

- Sequence

  When only two digits are used to represent a year, programs that collate year data will sort that data out of sequence in some cases For example, the year 2000 (if represented as 00) will be ordered prior to the year 1999 (if represented as 99)

- Data integrity

---

[1] Although IBM recognizes that the 21st century begins at 0000 hours, 1 January 2001, for purposes of this document, we are defining the 20th—21st century boundary to be between 2400 hours, 31 December 1999 and 0000 hours, 1 January 2000. This allows a discussion of the 21st century to include all dates with a 20yy format inclusive of the year 2000 Hence, the year 2100 is likewise relegated to the 22nd century

In programs where historical dates are used, for example all events occurring in 1800, 1900, and 2000 are not distinguishable when the years are represented by only 2 digits

- Leap year calculation.

  A specific year is a leap year if it is either evenly divisible by 400 **or** evenly divisible by 4 and not evenly divisible by 100 For example, the year 1900 was not a leap year but the year 2000 is a leap year Some potential exposures caused by the identification of the year 2000 as a non-leap year are

  - Day-in-year calculations The year 2000 has 366 days, not 365
  - Day-of-the-week calculations 28 February 2000 is a Monday and 1 March 1 is a Wednesday, not a Tuesday which is February 29, 2000
  - Week-of-the-year calculations. The 11th week of the year 2000 is 5 through 11 March, not 6 through 12 March.

Scope of Year2000 Transition

The computing environment today is mostly constructed and integrated around the theme of heterogeneous, open, distributed, and client-server computing The computers from different vendors are networked together to provide an integrated and distributed computing environment. Even though there are many different kinds of platforms in such an environment, they all tend to have the following common components

Computer Hardware

- CPUs
- storage devices
- input/output devices
- communication devices
- and so on

Computer Software

- Microcode
- Operating systems
- Middleware, such as database management systems, telecommunication control programs, transaction monitoring programs, and so on
- Programming languages, such as COBOL, Fortran, PASCAL, PL/I, C assembler
- Solution Developers and '3rd party' software providers
- Application programs

People

Those who develop and maintain the system, those who operate the system, those who provide its input and use its output, those who manage and ensure the quality of the system, and those who provide manual processing activities in a system These people include

- End users
- Management

- Auditors quality assurance people
- System analysts
- System designers
- Programmers
- Operations personnel

Data

The information that the system records and processes

Procedures

Formal policies and instructions for operating the system

Figure 1-1 illustrates these components in an organization's networked computing environment At each computing node (viewing the figure vertically), information systems (IS) personnel follow procedures to access the data by means of computer software and hardware The computing nodes (viewing the figure horizontally) are networked together by communication hardware and software to communicate to other computing nodes internal and/or external to the organization The data is thereby accessible anywhere and anytime while the computing nodes are connected

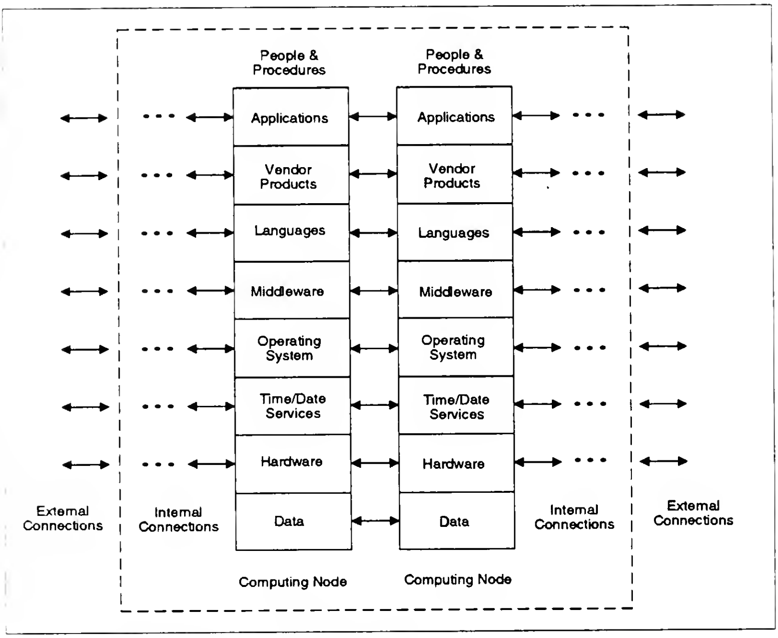

People & Procedures
People & Procedures
Applications
Applications
Vendor Products
Vendor Products
Languages
Languages
Middleware
Middleware
Operating System
Operating System
Time/Date Services
Time/Date Services
Hardware
Hardware
External Connections
Internal Connections
Data
Data
Internal Connections
External Connections
Computing Node
Computing Node

Figure 1-1 A Corporate Networked Computing Environment

The Year2000 phenomenon potentially has both vertical and horizontal impact on such a computing environment as pictured above Vertically, the Year2000 phenomenon can originate from or affect the key components of the computing systems, that is, hardware, software, people, data, and procedures Horizontally the phenomenon can propagate as the contaminated data flows to other computing systems inside and outside of the organization. In short, the scope spans the entire information technology industry

Although this is a complicated and far-reaching problem, it is not a technically difficult problem to resolve when viewed on a module-by-module or routine-by-routine basis The degree of complexity is directly related to the inter-relationships between routines and programs and the data passed between them **This is not a trivial programming exercise**. Your realization of a Year2000-ready system demands immediate management commitment, dedicated resources (personnel and hardware), strategic planning, rapid development, acceptance testing, a well-planned migration, and an adequate budget The following chapters address these issues and provide basic approaches for some specific issues

Chapter 2. Planning to Resolve Your Year2000 Exposures

To successfully address the challenges present within your computer system, you need to obtain your management's understanding, support, and an uncompromising commitment to provide resources to meet your needs Expect the need for a knowledgeable executive sponsor to address budgetary, personnel and hardware resource requirements, cross-department and cross-divisional requirements, and overall scheduling and project management It is imperative that this effort begin with and be managed through a central focal point responsible for critical project aspects such as overall scheduling, coordinating, and setting a consistent methodology through all project phases

This chapter and the following focus on those project phases planning, exposure identification, exposure elimination, testing, migration, and the selection of available tools The larger your computing environment, the more diverse its software, the more decentralized its physical environment, the greater control you must exercise, and the greater the communication that must exist across the individual projects

The time to begin both planning and your Year2000 transition is now. Consider the following

- Getting requirements and design changes into the development cycle takes time The review and modification of the application takes time Securing resources and skills takes time Handling the problem in real time will disrupt your customer services, and the business impact will be significant
- Most organizations are already short-handed when addressing their current workload and the challenges they possess Therefore, with this additional effort, perform a risk assessment and identify what is critical to the success of your business and determine and prioritize those work items
- A significant amount of code rework may be required to complete your Year2000 transition. It is not merely a problem that can be fixed by expanding the data fields You must make changes to your data dictionary, data bases, files, programs, and so on
- Within some institutions, programs are already producing incorrect output, and many organizations that aren't today, can expect problems tomorrow For example, insurance companies, when calculating rates for persons born in '94 might find themselves assessing 101-year-old rates on a new-born scale or potentially assessing a new-born at the same rate as a centenarian

For some organizations, it may not be mandatory to act now, but it could save code redesign, especially data rework. Each day that passes prior to your year 2000 transition allows more data to be added to your data bases and the potential for additional routines and programs that are added to your system Also, expertise that is present today will no longer be available later.

Finally, there have been projections that the availability of consulting and outsourcing services to meet Year2000 transition needs will become increasingly limited as we approach the year 2000. For organizations with applications that handle future dates or those with thousands of application programs, the consequence of delaying the resolution of the problem could be disastrous to their continued future success

© Copyright IBM Corp 1995, 1996

Planning Considerations

You need to plan for changes across all aspects of your IS environment The following task categories might prove useful when approaching this task

1 Identify and communicate the organization goal

   The goal is to have the function and operation of an organization Year2000-ready before any disruption caused by 2-digit-year data occurs **Year2000-Ready** means that products, programs, files, databases and processes have or produce no logical or arithmetic inconsistencies when dealing with dates beyond 1999

2 Identify the deliverables and associated schedules for the following

   - Hardware
   - Software
   - Documentation
   - Training
   - Maintenance
   - Operations
   - Administration
   - Acceptance criteria for all deliverables

3 Analyze job assignments

   For each task,

   - Identify the responsible person or organization, for example
     - Customers
     - Management
       - Chief Executive Officer
       - Chief Financial Officer
       - Chief Information Officer
       - Software Development Managers
       - Operation/Administration Managers
       - Budget/Finance Managers
       - others
     - Computer vendors
     - Solution Developers
     - IT outsourcing vendors
     - Consulting/Integration services providers
     - System analysts
     - System designers
     - System/Application Programmers
     - Operations personnel
     - End Users
     - Auditors, quality assurance people
   - Measure the estimated completion time
   - Identify precedences/dependencies
   - Identify resources and skills
   - Identify critical path schedule
   - Measure efforts
   - Measure costs
   - Identify technical factors
   - Analyze potential benefits
     - Return on investment
     - Achievement of business goals
     - Potential quality and acceptance of the approach

  - Your business keeps running
- Analyze risk factors
  - Complexity of the task
  - Resource/time constraints
  - Length of project
  - Critical development skills

4. Measure the criticality of each task and prioritize

   Evaluate and determine how critical the functions of each entity are to the business success of the organization, and prioritize the sequence of providing Year2000-ready solutions. An entity could be an individual, a department, a division, a business unit, customers, vendors, and so on, that are involved in the operations of the organization. The factors contributing to how critical you consider a task to be might include pressure of demand from end users for Year2000-ready systems, legal issues, financial issues, or political issues.

   Impact to Business

   To determine the impact to your business, consider including these tasks:

   - Critical to the operation of the business (such as legal compliance)
   - Critical to the uninterrupted operation of the business (such as payroll)
   - Required to support the business (such as management and financial reports)
   - Required to support the business, however, the importance and timetable for the activity is lower than an item above (such as regular scheduled reports)
   - Desirable, but not absolutely required to support the business

   Impact to Operations

   Once classified by task, then determine impact severity. For example, you could use categories such as:

Fatal	Operations will ABEND or terminate
Critical	Operations will produce an incorrect result. (For example, expiration dates for food or pharmaceutical products are calculated as over 100 years old, not one or several days old.)
Marginal	Minor inconvenience, annoyance, or irritation. (For example, inventory reports collate dates of 00 prior to 99.)

   Based on the impact to a particular process, evaluate the desirability of reworking a particular piece of code. Here is an opportunity for your IS management and your business strategists to affect overall business practice. Together these groups should consider possibilities such as:

   - Abandoning the business process
   - Combining the process with other processes
   - Replacing the process with a new state-of-the-art process

5. Establish a 'critical event horizon'

   Business environments are unique. The initial date your institution will begin experiencing Year2000 problems is also unique. If you prepare business forecasts of a 3-year cycle, the fourth quarter of 1996 might be your critical event horizon. If you deal in automobile loans, 1995 might be your critical event horizon It is likely to be a very rare institution that will not experience some form of Year2000 difficulty until 1999 or 2000

6 Provide data administration

- Identify the scope and responsibility of migrating the affected data
    - Exclusive The affected data object is created and processed exclusively by this business area and is independent of any other business area This could be at an individual, department or the business area level with further decomposition and analysis
    - Primary responsibility The business area defines the affected data object and other business areas should use that definition or negotiate for its redefinition
    - Secondary responsibility The affected data object is defined and created by a different business area in the enterprise, and is distributed only within the scope of the enterprise Each business area defines its own use of the object which is provided by the business area with the primary responsibility
    - External exposures
        - The affected data object is defined and created by either this or a different business area in the enterprise, and is distributed beyond the scope of the enterprise
        - Data is created outside your enterprise and then imported and used within it
- Determine formats of the data dictionary
- Determine procedures for changing and entering data elements
- Determine procedures for collaborative data sharing and use

7 Decide project technical and management approaches

- Programming standards, conventions, and guidelines
- Platform for application development
- Hardware/software
- Development methodology
- Development and test procedures
- Prototyping and parallel development
    - Commonly used in software development projects and should be applied wherever appropriate
    - Apply a divide-and-conquer approach to partition the Year2000 project into smaller projects so that development and testing can proceed in parallel
    - Parallel development can shorten the development cycle This is extremely critical when dealing with time-sensitive projects such as this Year2000 project
- Process/data modeling
- Data dictionary
- Documentation structure, layout, and standards
- Reviews and walk-throughs

  - Quality assurance procedures
  - Testing methodology
  - Automated tools
  - Migrations of and bridgings to existing Year2000-ready systems
  - Estimated future costs of maintenance

8. Identify project constraints, interfaces, and dependencies
  - End users/customers
    - Availability of test and other data
    - Availability of facilities and services
    - Responsibility for reviews
    - Responsibility for end user tests
    - Other actions
  - Special contract negotiations
  - Outsourcing and consulting services
    - Justification for the outsourcing and consulting services
    - Obligations for the outsourcing and consulting services providers
  - Interfaces and dependencies with other projects
  - Supporting services and facilities required
  - Hardware and software to be used
  - Solution Developer-automated tools to be used
  - Risks and alternative solutions
  - Other assumptions

9. Provide standards, guidelines, quality assurance, and review procedures
  - Year2000-ready standards for purchasing of hardware/software vendor products
  - Year2000-ready system requirements on request for proposal for outsourcing and integration services providers
  - Organizational Year2000-ready standard/guidelines/process for specification, design, development, and testing of new and existing software
  - Year2000-ready checklists for potential exposures
  - Standard for machine-human interface (Refer to "Guidelines" on page 4-12 for a list of standards.)
  - Procedures for submitting and processing proposed changes. The procedures should evaluate why change is needed, consequence of not making the change, and its effect on product, cost, and schedule

- Procedures for sign-offs and approvals The procedures should solicit comments from knowledgeable and affected people about likely effects on product, documents, schedule, and costs
- Procedures for future follow-up

Inventory Your Software Portfolio

Once you have put your Year2000 plan in place, you can then begin the task of converting your software programs for Year2000 readiness

The first step requires that you thoroughly understand your computing environment and compile an inventory of all the software programs within that environment Such an inventory allows you to

- analyze your portfolio for definition and movement of date-related data elements and the use of date-related calculations and manipulation
- identify and remove Year2000 exposures
- track and control changes to your portfolio to more easily monitor and prevent injecting new Year2000 exposures into your inventory while your Year2000 resolution work progresses
- test the new (Year2000-ready) version of the software programs in your portfolio

Once you have completed the above activities, you are ready to migrate from your current computing environment to your Year2000-ready environment The following chapters discuss these activities in detail and provide options and suggestions for your consideration

Chapter 3. Identifying 2-Digit-Year Exposures

To identify the potential exposures caused by using 2-digit-year representations of dates, you first need to locate references to all date-related data.

Locating References

Locating date-related data and code is itself a major piece of the work effort that you must address. The most complete method starts with an inventory of every programming entity used in your IS center. Once compiled, you can review each program individually. Alternatively, use the following, more systematic, approaches to locate 2-digit-year data and date-related code. (Solution Developer products are available to assist this effort. Refer to "Solution Developer Tools" on page 7-71 for a list of those Solution Developers and their products.)

1. Review the following documents for direct or indirect references to date-related data and formats. Then trace these references back to the application source code to locate references in that code as well.
   - Requests for proposal
   - Statements of work
   - Planning documents that describe future IS needs
   - Existing studies about the current system
   - Software development standards and process documents
   - Software quality assurance requirements
   - System requirements specifications
   - System design specifications
   - Program specifications
   - User instructions and procedures
   - Data dictionaries
2. Review program information for date references to include, for example, date variables, date functions or routines, and character strings. Character strings might include the following in either your code or its comments:
   - AS-OF, ASOF
   - BEGIN, BEG, BGN
   - CCYY
   - CYYDDD, CYYDDMM, CYYMMDD
   - CURR, CURRENT
   - DATE, DAT, DTE, DT
   - DAY, DA, DD
   - DDMMYY, DDDYY
   - DIFFDATE
   - DOB
   - DOH
   - END
   - EXPIRE, EXP
   - JULIAN
   - MONTH, MON, MO, MMM
   - MDY, MMDDYY
   - START
   - TERM
   - TIME
   - TIMESTAMP, TIME-STAMP
   - TIMEDATE
   - THISDATE
   - TOD, T-O-D
   - WEEK
   - WEEKDAY
   - YEAR, YR, YY
   - YMD, YYMMDD, YYDDD
   - and so on
   - Data entry forms, screen display formats, report formats
   - Definitions of data fields, records, structures, files, and databases

© Copyright IBM Corp 1995, 1996

- Source code, computer program listings, cross-reference reports
- Command languages, for example, JCL, REXX, CLIST, EXEC, and CL
- Data indexes and catalogs, table sizes
- Data dictionaries
- Date/time service routines
- Sort routines

3. Use a test system

   Install an isolated, non-production system with a duplicated image of your system and application software. In large systems, this could be an LPAR (logical partition of the mainframe), or for other platforms consider dedicating a separate machine, segregated (in either time or place) from any other system(s). This segregation is crucial to guarantee that you will avoid contamination due to system clock advancement or untested, and yet imperfect, modified code.

   - With a changed date/time setting

     Set the system date and time to a future date and time value after 1 January 2000. During such testing, be certain to use compatible data that is synchronized with the revised application software, that data crosses a 100-year boundary, and be certain to update your current operating procedures to reflect this new data requirement as well. For example, the 100-year window could be 1900-1999 or 1995-2094, or you could use both range types. The more varied are your window type(s), the more incorrect code you will uncover. This testing will help you identify many (but not all) Year2000 exposures.

   - With changed data

     If you only change date fields in a routine, you are not introducing new logic into that routine. Although the fields are increased from 2-digit to 4-digit fields, you only need to recompile the routine to generate those new field lengths. If you didn't change the logic, you needn't test that logic. Typical testing is appropriate in a separate test system using new data, but Year2000-specific testing is really unnecessary. In this testing, ensure that the changes in the operand lengths produce the expected results. If the results match those produced prior to your changes, then the application is successfully performing data calculations using the new 4-digit data rather than the previous 2-digit data, and you have met you migration criteria.

   Refer to Chapter 5, "Testing Techniques for Year2000 Changes" on page 5-1 for a more detailed discussion of testing techniques and issues.

4. Use a combination of the above approaches.

Tracing References Back to Their Source

For any potential exposure identified, identify all direct and indirect references of this exposure. You can do so by tracing the flow of the data to identify its immediate source and destination and then repeating the tracing process until all sources and destinations of all potential data exposures are identified.

Determining the Impact of 2-Digit-Year Data Fields

Once you have located date-related data fields by one or more of the above approaches, you can classify the use or reference of those fields into one of the following categories

- No Impact
  - The program uses a 4-digit-year representation in all occurrences
  - The program uses a 2-digit-year representation within a program, but does not have any internal exposures, nor does it externalize the 2-digit year in any way
  - The program uses a 2-digit-year representation within a program, but does not have internal exposures The 2-digit year is externalized but can not be referenced (for example, for display-only) by another program.

  Note: Such output might be labelled as 'cosmetic only' and for interpretation by a human only, but this too might have its own set of ramifications

  - A municipality that tracks school-age children will more frequently begin 'inviting' centenarians to enroll in kindergarten If a printout of residents reads, `Birthdate: 10/14/89`, what will the clerk compiling the list of 6-year-old children assume?
  - Another type of exposure is externalized by 'display-only' dates that have been coordinated with a hard-coded '19' for the century Such 2-digit exposures might exist for terminal display or special forms where the '19' is pre-printed Therefore, be aware of, and consider, potential impact of date fields in all situations, these might not always be obvious exposures
- Impact
  - The program uses a 2-digit-year representation within a program It does not have internal exposures, but the 2-digit year is externalized and could be referenced by another program.
  - The program uses a 2-digit-year representation and has internal exposures

Investigating How Other Software Entities Use the Data

You also need to investigate the ways data is shared among software entities The greater the degree and scope of data sharing, the more global is your task and the more critical is the need to prevent the further propagation of and data 'contamination' by 2-digit-year data Except possibly in a rather small IS environment, you cannot know how output is used by all other programs that might access it Several factors that affect how data is shared or used among software entities follow These factors and the types of data sharing provide some of the links making up this data 'web'

Data Sharing

Three factors affect the sharing of data between modules.

1. The number of data items passed between modules (The more data passed, the tighter the relationship.)
2. The amount of control data passed between modules (The more control data passed, the tighter the relationship.)
3. The amount of global data elements shared between modules (The more global data elements shared, the tighter the relationship)

Several types of data sharing can occur between two modules in a program Two modules can communicate

- Through a variable or data structure (for example, array, table, or record) that is passed directly as a parameter between the two modules The data is used for problem-related data processing not for program control purposes.
- Through a variable or data structure (for example, array, table, or record) that is passed directly as a parameter between the two modules However, only part of the data in these composite data elements is needed in the call, that is, more data is passed than needed. A change in one of the data structures to accommodate a change in either the calling or called module can affect other modules as well.
- By passing data from one module to another to control the order of instruction execution. (For example, a control flag is set in one module and tested in a CASE or WHEN statement in another module)
- By passing data between modules through some mutually agreed upon location in a global data area (for example, FORTRAN COMMON and PL/I EXTERNAL features) A change in one module might then require changes in other modules sharing the same data area.
- By one module reaching into the internals of another module to get or deposit data or control its function (For example, a branch from one module's code into another module A change in either module might require a thorough analysis of the internals of both modules to determine how to deal with the consequences of the change)

If contaminated (2-digit year data) is shared among modules, you must identify the exposure caused by the data sharing on the receiving side of the transaction

Once you have identified Year2000 exposures, apply the appropriate techniques (as provided in Chapter 4, "Reformatting Year-Date Notation" on page 4-1) to reformat these date and time representations.

Chapter 4. Reformatting Year-Date Notation

This chapter provides a number of techniques that you can employ to correct improper date notation and use Because some techniques are appropriate only to unique situations, this section also lists the advantages, disadvantages and IBM recommendations for their use.

When selecting a proposed Year2000 solution, evaluate the following factors

1 What is the external impact due to incompatible date format changes?

   That is, what other programs or what output will be affected and to what extent will those programs require change if this solution is implemented for this particular program?

2 How current are the program modules that reference the date formats externalized by the exposures?

   That is, are there any plans to either eliminate or replace this particular program or routine, the programs that input to it, or those that receive or use its output?

3 What functions will be impaired due to Year2000 exposures?

   That is, will any mission-critical function within your company be compromised due to not reworking or replacing a particular program?

Solutions and Techniques

As you identify Year2000 exposures by the approaches described in Chapter 3, "Identifying 2-Digit-Year Exposures" on page 3-1, your next step is to rework the current program and data exposures to make your applications Year2000-ready You can apply the following solutions to remove potential Year2000 exposures Each solution is presented with an example technique to change the potential exposure These suggested techniques require both program and data changes Several solutions and techniques and their associated pros and cons follow

Solution #1: Conversion to Full 4-Digit-Year Format

This solution is a 4-digit solution that externalizes a 4-digit-year format

This approach requires changes to both the data and the programs by **converting all references and/or uses** of 2-digit-year format (YY) to 4-digit-year format (YYYY). It also requires that you convert all software programs that reference or use the updated data simultaneously, or use a bridging mechanism to perform the conversion between old and new data and programs Refer to "Bridge Programs Help Stage Format Conversions" on page 4-10 for more details on bridge programs. (You can accomplish this program and data conversion in steps.) Otherwise, you will immediately encounter data integrity problems caused by the inconsistency of date/time data formats

To ease your migration, you might consider ignoring any non-impact (cosmetic) data fields in the YY format A cosmetic date is one, that if externalized, is only interpreted by humans. Such occurrences might include the date on an output separator page or a display-only date on a screen in a panel-driven application

Note: Be careful when selecting those situations that you decide to ignore and call cosmetic only Be certain that they will not cause any data integrity

© Copyright IBM Corp 1995, 1996

exposures or ambiguity or are not accessed by any other program Such instances of non-problem YY formats appear in a report header that shows the printing date of the report. The date is meant for human understanding only, not computer program manipulation. Consider the potential for future change For example.

- Today's reports might be written to a data set tomorrow
- Display-only dates today may prove useful as a collating value when archiving that output tomorrow to meet a new business or government standard
- Even when viewed by a human, 2-digit dates can prove ambiguous if the data spans 100 years

If you allow the end user to continue to input 2-digit dates for compatibility and ease of data entry, then the responsibility to translate that data into a full 4-digit date falls to you, the application or systems programmer One possible solution is to apply a context-sensitive prompt to allow the user to select a century indicator For example, allow all dates to be entered as 2-digit dates and automatically prefix those with the current century unless the date is a future date or historical date What constitutes "future" or "historical" is your decision but could be any date other than today's current day, week, month, year, and so on Using this scheme, a future date in context of a loan maturity date could be set to 20yy, or a historical date automatically forces the user to select a century from a 'choose a century' (16, 17, 18) prompt list.

Pros and Cons

Pros

1. Can provide 4-digit-year format It is considered to be the **only** complete, permanent, and obvious solution
2. Provides increased security against potential inappropriate decisions today if you do not selectively ignore 'cosmetic-only' situations.
3. Can ease your migration if you selectively ignore 'cosmetic-only' situations.

Cons

1. Need to convert the year data from 2-digit format to 4-digit format in all cases
2. Requires that you relocate adjacent fields in the date field layout, and usually requires that you increase record lengths
3. Inherent future risk in initial assessment that determined a particularly situation can be ignored as 'cosmetic only'
4. Increased DASD space usage required due to data field expansion of data (consider including not only active but also archive data) and duplicate DASD space during conversion
5. Might experience a performance impact due to increased time in processing and date access
6. Some programming languages allow integer dates that are offset from a base date to be stored in files, data bases or passed as parameters between programs. Such integer dates provided by COBOL intrinsic functions, Language Environment callable

services, the CICS FORMATTIME command DAYCOUNT option, and other similar functions must conform to the standard YYYYMMDD format. This standard eliminates potential ambiguous data and errors due to each integer-date system using a unique starting date. Therefore, the potential for mixing incompatible integer dates when passed outside a single source module is extremely high and must be avoided

Solution #2: Windowing Techniques

This is a 2-digit solution that externalizes either 2-digit or 4-digit-year formats This approach requires changes to your programs only; no data changes are required

CAUTION:
These approaches can be applied only to dates within a maximum 100-year period at any one time. This solution is considered temporary because there is no guarantee that in the future, your applications will not expand to process dates that are more than 100 years apart. Therefore, this approach always carries with it a potential future exposure. (For example, humans are living longer. Therefore data bases that include birthdays (medical, civil, insurance, and so on) and the applications that access that data are already at risk with many dates spanning 100+ years.)

Two types of **windowing** techniques have been defined. the fixed window technique and the sliding (rolling) window technique.

Fixed Window Technique

The **fixed window** technique uses a static 100-year interval that generally crosses a century boundary This technique determines the century of a 2-digit year by comparing the 2-digit year against a window of 100 years. The user specifies the number of years in the past and future relative to a specific year within the 100-year interval.

Consider this specific example if the years of date-related data of your application fall in the range of 1 January 1960 to 31 December 2059, you can use a 2-digit year to distinguish dates prior to the year 2000 from the year 2000 and beyond If using the current system year of 1995, the number of years in the past and future are specified as 35 and 64, respectively Program logic determines the century based on the following data checking If the 2-digit year representation of a specific year is xy then if:

- xy≥60, then it is a 20th century date (19xy)
- Otherwise (that is, xy≤59), it is a 21st century date (20xy)

If, for example, you need to maintain a window of 35 past years and 64 future years, such that next year, 1996, your application can successfully deal with dates in the range 1961 through 2060, you need to adjust this program checking every year The inherent future risk when employing this technique is obvious, and when compared to the sliding window technique is far less desirable

Pros and Cons

Pros

1. No need to expand the 2-digit-year data to a 4-digit format.
2. Can provide 4-digit-year format for data reference
3. Can distinguish years from different centuries using only 2-digit-year format (provided the years being processed are in the range of 100 years at any one time).
4. Can be useful if the particular program is being phased out, and a temporary solution is appropriate

Cons

1. Potential exposures exist when/if the function of the software application needs to process years beyond the range of 100 years
2. Expect a performance impact in direct proportion to the quantity of date processing the particular application handles due to the overhead of 2- to 4-digit-year conversion
3. All programs that use the fixed window technique may need to be manually updated on a yearly basis depending on how your date routine is packaged
4. All programs that accept output from the fixed window technique must use the same assumptions (current date, past and future windows)
5. Retaining a 2-digit year representation does not provide collating sequence support Nor does the use of a fixed window technique provide indexing sequence support when 2-digit years are used as index keys in indexed files You will need to provide additional processing to obtain correct collating and indexing sequence output

Sliding Window Technique

The **sliding window** technique uses a self-advancing 100-year interval that generally crosses a century boundary This technique determines the century of a 2-digit year by comparing the 2-digit year against a window of 100 years The user specifies the number of years in the past and future relative to the system year (generally the current year) that the system sets[1] and maintains Your applications can access the date that the system sets and automatically advances This is the main advantage of using a sliding window over the fixed window (where the window is immovable without manually revising the programs each year)

As appropriate to your application environment, you can maintain more than one window For example, you could set one window to process historical dates, one for mortgage dates, one for birth dates, and so on; and the program adjusts the system date and past and future windows to meet the specific application's needs

---

[1] IBM product, Language Environment, provides an option whereby you can set the system year to other than the current year This flexibility then allows you to set a 100-year range you require, it need not even contain the current year. Refer to Chapter 7, "Tool Categories and Available Tools to Ease Year2000 Changes" on page 7-1.

Consider this specific example. If the dates in your application fall into a range of 35 years in the past and 64 years into the future, based on the current year, 1995, your program can accept and accurately deal with dates of 1960 through 2059 Next year, 1996, the window advances and your application accurately deals with dates of 1961 through 2060

Graphically, Figure 4-1 on page 4-5 illustrates this example using the current (1995) 100-year window and that same window when the current system date has progressed to the year 2024

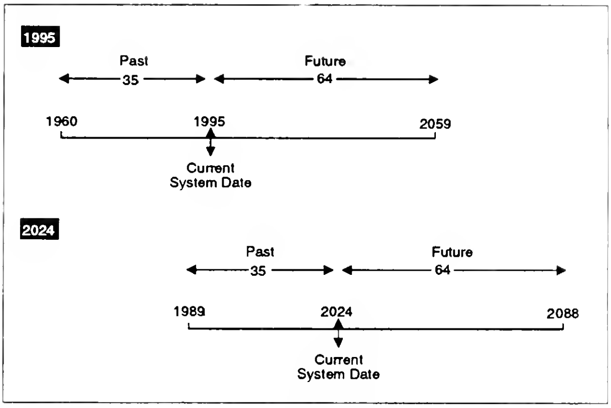


Figure 4-1 Graphical Representation of the Sliding Window Technique (Using two current system dates, 1995 and 2024, as an example.)

A sliding window approach requires programming logic to interpret the meaning of all 2-digit year data Such additional programming logic could be packaged into a common data/time service routine, callable from a 2-digit year data exploiter This would reduce the programming overhead and impact to the calling programs IBM product, Language Environment, provides common date/time service routines with sliding window features By default, Language Environment uses a window of 80 years in the past and 20 years into the future that automatically adjusts based on the current year date. For details on using and setting the past/future window range, refer to Chapter 7, "Tool Categories and Available Tools to Ease Year2000 Changes" on page 7-1. For an example of how DFSORT/MVS is implementing a sliding window to sort, merge, and transform 2-digit year data to 4-digit year data, refer to "DFSORT" on page 7-23

Pros and Cons

Pros

1. No need to expand the 2-digit-year format to a 4-digit format
2. Can provide 4-digit-year format for data reference
3. Can distinguish years from different centuries using only 2-digit-year format (provided the years being processed are in the range of 100 years at any one time)

4. No need to convert the date data to a new date representation scheme

Cons

1. Potential exposures exist when/if the function of the software application needs to process years beyond the range of 100 years
2. Potential performance impact in direct proportion to the quantity of date processing the particular application handles
3. All programs that accept output from the sliding window technique must use the same assumptions (current date, past and future windows)
4. Retaining a 2-digit year representation does not provide collating sequence support. Nor does the use of a sliding window technique provide indexing sequence support when 2-digit years are used as index keys in indexed files. You will need to provide additional processing to obtain correct collating and indexing sequence output

Solution #3: A 2-Digit Encoding/Compression Scheme

This is a 2-digit solution that externalizes only a 2-digit-year format. It requires changes to both your data and your programs. It also requires that you convert, simultaneously, all applications that reference or use the updated data

Example techniques that are useful when using this solution include **encoding** or **compressing** 4-digit-year data into 2-digit existing space. This section presents several specific examples, many others exist and might prove more applicable to your specific needs.

CAUTION:
Apply this approach with caution. It is considered to be the least desirable approach and should only be used if absolutely necessary. Be certain that the new encoding or numbering scheme does not affect the proper functioning of your programs after all the data changes are implemented.

This solution is considered temporary because there is no guarantee that in the future, your applications will not expand to process dates that are outside the encoding limits.

Some examples include:

- **Example 1:** Convert the numbering scheme from decimal to hexadecimal. However two hexadecimal digits can provide only a maximum of 255 years, for example.

  ```
  D'1900' + X'FF' = 1900 + 255 = 2155
  ```

  Specific date conversions might be

  - Convert the year 1900 represented by D'00' to X'00'
  - Convert the year 1999 represented by D'99' to X'63'
  - Convert the year 2000 represented by D'00' to X'64'

- **Example 2:** Convert the data type from the 2-byte character representation of the 2-digit year to a 1-byte **unsigned packed decimal** (two digit) representation, and use the freed byte to append two **unsigned packed decimal** digits to represent a 4-digit year. For example.

  - Convert the year 1900 (represented by character string '00' (= EBCDIC X'F0F0')) to unsigned packed decimal X'00' and prefix unsigned packed decimal X'19' in front of X'00' to yield X'1900' in unsigned packed decimal
  - Convert the year 1999 (represented by character string '99' (= EBCDIC X'F9F9')) to unsigned packed decimal X'99' and prefix unsigned packed decimal X'19' in front of X'99' to yield X'1999' in unsigned packed decimal
  - Convert the year 2000 (represented by character string '00' (= EBCDIC X'F0F0')) to unsigned packed decimal X'00' and prefix unsigned packed decimal X'20' in front of X'00' to yield X'2000' in unsigned packed decimal.

- **Example 3**: Convert the numbering scheme from decimal to a user-defined numbering scheme. The mapping between the new and old schemes can be defined by a table or mapping function; and the conversion between the two numbering schemes can be done by table lookup or functional mapping. Figure 4-2 on page 4-7 presents one such possible user-defined table that provides for values up to 1295 within a 2-digit field. This scheme uses the characters 0-9 and A-Z to represent decimal values 0-35, respectively. This base 36 notation is thereby capable of extending the hexadecimal example on page 4-6 by 1040 more years. For example

  0'1900' + base$_{36}$'ZZ' = 1900 + 1295 = 3195

Figure 4-2. Example User-Defined Date/Year Conversion Table

2-Character Year (Encoded) Value	Converted Data/Year Value	Year (When Using 1900 as the Base Year)
00 - 0Z	00 - 35	1900 - 1935
10 - 1Z	36 - 71	1936 - 1971
20 - 2Z	72 - 107	1972 - 2007
30 - 3Z	108 - 143	2008 - 2043
40 - 4Z	144 - 179	2044 - 2079
R0 - RZ	972 - 1007	2872 - 2907
Z0 - ZZ	1260 - 1295	3160 - 3195

- **Example 4**: Pack a 3-digit date field with a 4-digit year date by the use of the CYY format. Using a conversion table or offset, you can indicate, for example, that C=0, 1, or 2 represents 19, 20, or 21, respectively. When your application appends the C to the YY field, your system produces full, 4-digit year dates, the range of which depends on the conversion mechanism. Using decimals 0-9 to represent 19-28, this scheme provides a solution from 1900 through 2899, but it is likely to require end-user procedural changes, education, and typical learning curve time and errors.

  One advantage to setting C=0 to represent 19 and so on is that it might provide a compatible, non-disruptive change to some existing application routines if such a field is currently prefixed to your YY data field and set to 0.

  A variation on this same scheme would include the use of the CCYY format where the CC can be used to represent the actual century indicator, 18, 19, 20 and so on, or an encoded value for example, "00", "01", and "02", to represent 19, 20, 21, respectively (see "Solution #1 Conversion to Full 4-Digit-Year Format" on page 4-1)

When adding either the `C` or `CC` prefix to the `YY` field for `CYY` or `CCYY` representation, `C` or `CC` can be extracted from a separate field, one that is not necessarily adjacent to or preceding the `YY` field This then can relieve any restrictions you might currently have due to your date field length It does, however, require further programming logic and data manipulation

Pros and Cons

Pros

1. No need to expand the 2-digit-year data format to 4-digit data format (For example, there is no need to increase the fields in data bases and tables to accommodate dates above 99 which would increase DASD usage.) Further, this saves the effort that would be required to rebuild your database(s)
2. Can distinguish years from different centuries using the 2-character-year format
3. If you use a COBOL COMP-3 format, you can pack a CCYYMMDD date into an existing 6-byte field (with one byte left over) This technique allows you to retain the original field size and eliminates your need to relocate adjacent fields Applications that use the data for calculations run faster because the data is already packed
4. If you use a flagged Julian format (CYYDDD) where `C` is used as the 'century indicator', the format does not require expansion of the date field

Cons

1. Depending upon the choice of data representation you implement, this scheme can be applied only to a limited date range. For example, you are limited to 255 years when using hexadecimal representation
2. All programs that use this scheme and need to access the output of the 2-character conversion must change simultaneously
3. Due to data conversion (calls and processing) you might experience a performance impact in direct proportion to the quantity of date processing the particular application handles.
4. Depending upon the choice of data representation you implement, you might experience incorrect data sequencing if you do not add further programming logic
5. Encoded dates require conversion whenever you work with that data Therefore, the presence of encoded dates will add another layer of complexity to such tasks as problem determination.
6. You must convert the data before it can be displayed in Gregorian format, and some encoded data can only be viewed in hexadecimal format. This is both impractical for human reading and also impractical or impossible to print.

Using a Common Date/Time Service Routine

In large system applications it is common to find that more than one date/time service is in use. However, some date/time service routines may have Year2000 exposures of their own, for example, the routine(s) only provides a 2-digit-year format. Fixing the exposed date/time routine(s) is one possible solution. Selecting a vendor date/time routine that is certified as Year2000-ready for consolidation and/or replacement of your 'in-house' date/time service routines is another alternative

While fixing your current date/time routine(s) exposures, you may find it worth your effort to consolidate all your date/time service routines into one **common date/time service routine** If you then detect any Year2000 exposures during or following the consolidation, you can reformat your program and data and decide on the appropriate solution(s) you will use That is, you can package any new code, encoding and conversion routines, windowing-specific data, and so on into the common date/time service routine. This common date/time service routine package might then be considered a 4-digit solution that externalizes both 2-digit and 4-digit-year formats. The benefit of using such a common date/time service routine is lower future maintenance because all services are consolidated rather than replicated throughout your applications.

Considerations When Selecting Solutions

As described in "Determining the Impact of 2-Digit-Year Data Fields" on page 3-3, potential 2-digit-year exposures can be classified into two categories (no impact and impact) When selecting an appropriate solution(s) for the 'impact' categories, be certain to consider not only the applicability of the solution(s) on the module itself but also the potential impact and adjustments on the external modules that receive data from this module. You have three basic choices, you can

- Change your application
- Change your application and the data
- Invest in a new application (which could also require some date data changes)

Certainly, most IS organizations will build their 2000-ready system on a combination of these choices. When more than one solution appears feasible, weigh its appropriateness based on.

- Time available
- Resources available (personnel and hardware)
- Project cost (individual application conversions and overall)

As today's IS environment becomes increasingly more complex and sophisticated, the instances of program and data isolation decreases Networking, open and distributed computing allow data to flow from site to site, system program to application program (or application program to system program), and so on. You must ensure that these layers of software 'speak the same language' As you add in-house code, Solution Developer-written code, and migrate your operating system, be certain to review that software for date format compatibility

Solution Applicability

Different combinations of solutions are applicable to different situations Evaluate solutions based on a 'best-solution combination' basis when considering both a module itself and other related modules For example, when applying

- **Solution #1** (full 4-digit solution) to a certain module, another module that receives data from this module could receive
  - 2-digit-year data as before, provided there is no exposure for itself or
  - 2-digit-year data as before and apply Solution #2 (windowing techniques) for its own exposures or
  - 4-digit-year data and apply Solution #1 (full 4-digit solution) for its own exposure removal.
- **Solution #2** (windowing techniques) to a certain module, another module that receives data from this module may either receive 2-digit-year data as before and apply Solution #2 itself or receive 4-digit-year data and apply Solution #1 (full 4-digit solution) for its own exposure removal
- **Solution #3** (the encoding or compression technique) to a certain module, another module that receives data from this module may need to apply Solution #3 as well to maintain data consistency with the data representation scheme Another alternative is to apply Solution #3 to the impacted module, and then convert that 2-digit-year format to 4-digit-year format before externalizing that data to another module This receiving module can then proceed with 4-digit data, and if necessary, apply Solution #1 (full 4-digit solution) to adopt the 4-digit-year data for its own exposure removal

Bridge Programs Help Stage Format Conversions

Bridge programs are often used to convert data from one record format to another If you use such a program, it should define the.

- Input date format and encoding method
- Output date format and encoding method
- Logic that converts the data from input format to output format based on their encoding methods.

You can apply bridge programs during program execution or file and/or database conversion For application during program execution, the conversion occurs each time data is passed between programs or between program and source data using different record formats For application during file and/or database conversion, the bridge program reads one record at a time from the source, transparently converts the record format, and writes out the data in the new format to the destination The process is incremental and can continue until all the records in the source are converted

Bridge programs for data format conversion provide the following benefits

- Granularity when changing the code and/or data

  With the scope of the Year2000 project, it is not practical (if possible) to change all the code and data at once. Bridge programs allow the gradual conversion of the programs and/or data and still maintain the compatibility between different data formats For example, you can change some of your programs to adopt a new data format and still be able to communicate to programs using the old format (after conversion by the bridge programs)

Therefore, changes to the remainder of your programs can be performed in an incremental manner as convenient.

- Flexibility when choosing appropriate solutions

  Bridge programs allow you to select the appropriate mix of different solutions to best meet your specific circumstances while maintaining the compatibility between different data formats For example, you can design your programs so that they can process data in different formats You can then have active data in a 4-digit-year format and archive the same type of data in 2-digit-year format The bridge program distinguishes the data in these various formats by reading the records and, when necessary, converting the data to the appropriate format.

Other Programming Situations

Other programming situations you should consider might include

- The possibility that a data format has become outdated and will not function correctly beyond 31 December 1999 (or earlier)

  Such data formats might be outdated even earlier and have already been superseded by another method by the Solution Developer For example, The MVS platform will no longer support VSAM catalogs for processing when the system date is beyond 1999 To support data sets which need to have explicit expiration dates beyond the end of 1999, or to create cataloged data sets after 1999 on MVS systems, you must use ICF catalogs Using VSAM catalogs on the MVS platform (including OS/390) will no longer function, this requires a programming change to take advantage of the alternative solution

- When migrating to year2000 support, your applications (operations) might support only 2-digit-year format, only 4-digit-year format, or both formats

  It is possible that the 2-digit values are assumed to be 19xx dates Therefore, be aware that all these must be eventually updated or the functions will fail or will give unpredictable results after 31 December 1999

- Changes to operations procedures

  Be certain to educate your operators about command changes so that they know when they must use a full 4-digit date (for example, 2000, to avoid implying 1900 if they only enter 00).

- When erroneous data would be produced for a limited and known timeframe and changes are not justified

  You might have a situation that is best handled manually to meet a short time period where programming changes simply aren't justified Consider using 2-digit-year data if a timeframe such as a single 24-hour period (31 December 1999 to 1 January 2000) or a single week (25 December 1999 through 1 January 2000) would be the only time your application will not provide correct results. For example, a program that looks at a sales report to compare the current day's merchandise movement with the previous 7 days. Because there would only be 8 reports containing both 1999 dates and 2000 dates, you might decide to handle the problem manually rather than changing the code.

Note: Don't fail to use a certain amount of common sense when deciding what applications to change, which to replace, and which to ignore. Do not lose your perspective of your institution's business needs and priorities and the impact and cost a particular application's change might have on attaining those goals

Guidelines

While retaining a perspective of any external impact, module currency, and what functions are impaired due to Year2000 exposures, use the following guidelines when applying Year2000 solutions

Note: This is not intended to be an exhaustive guideline, but rather a foundation upon which to start your specific Year2000 date-data resolution

1. Establish an in-house 'date standard' Conformance to the ISO Standard 8601 listed below would be a valuable starting point The earlier such a standard is in place, the sooner your IS organization will avoid creating new date issues and the propagation of current ones. You can refer to
    - ANSI X3 30-1985 (R1991) *Representation for Calendar Date and Ordinal Date for Information Interchange*
    - ANSI X3 51-1994 *Information Systems – Representations of Universal Time, Local Time Differentials, and United States Time Zone References for Information Interchange*
    - ISO 8601.1988 *Data elements and interchange formats – Information interchange – Representation of dates and times*

    These standards and ordering information are listed in the section entitled "By Title" in Appendix C, "Bibliography" on page C-1
2. Minimize potential impact to external references due to incompatible date format changes For example,
    - Maintain the 2-digit-year format as an option when 4-digit format is required for an application program interface (API) that provides 2-digit-year data references
3. **Avoid** any ad-hoc solutions, such solutions inevitably require a future problem investigation and removal, and should be considered temporary solutions only For example
    - Do not determine the century of a 2-digit year by comparing the 2-digit year against a hard-coded threshold, for example, 60 If the 2-digit year is greater than or equal to 60, then the year is a 20th century year, otherwise, it is a 21st century year
    - Do not fix the leap year calculation formula by adding logic to check if the current year is the year 2000 This solution will temporarily fix the leap year calculation problem by singling out the year 2000, but it does not fix the leap year calculation problem for other years in the multiple of 400
4. A 2-digit-year format might be acceptable for human-only viewing purposes, for example, screen panels, hardcopy reports, and so on However, any such data can be, and often is, added to a data set and then read by another program A 'log' that can be used as input to any program should **not** be considered in this (non-impact, for human viewing only) category
5. When changing the date format of any 'log', ensure that all the contributing programs adopt the new date format as well.

6 Consider the Year2000 solutions listed in this document (see "Solutions and Techniques" on page 4-1) for applicability in the following order

- Using a common date/time service routine (a 4-digit 'solution' - that can support both 2- and 4-digit formats) This is
  - Considered a long-term solution
  - **The recommended solution** for its support of both 2- and 4-digit-year formats that provide a long-term solution and no impact to 2-digit-year data references
- Solution #1 (conversion to a full 4-digit-year format that externalizes 4-digit formats)
  - Considered a long-term solution
  - Only supports 4-digit-year formats that will have impact on 2-digit year data reference
- Solution #2 (windowing techniques that externalize both 2- and 4-digit formats)
  - Considered a temporary solution and **should only be used when Solution #1 is not practical** (This is an arguable issue, because there are applications that deal only with years in the range of 100 years However, there is no guarantee that the functions of the applications will never change in the future and then require 4-digit-year formats)
  - Has potential exposures when the function of the program needs to process years beyond the range of 100 years
  - Use this solution only when
    - Processing is always limited to the current date data, for example, at the time of IPL or time of job creation
    - Expanding the date-data field is costly, and the function of the software program will be phased out before any exposure occurs
- Solution #3 (2-digit encoding/compression scheme that externalizes 2-digit formats)
  - Has potential exposures when the function of the program needs to process years beyond the range that can be covered by the encoding or compression scheme
  - Should be used only when expanding the date-data field is costly and the function of the software program will be phased out before any exposure occurs

Chapter 5. Testing Techniques for Year2000 Changes

Testing can be formalized into a 4-phase process, as follows

Test Type	Used to Test
Unit testing	A single program module
Integration testing	A related group of program modules
System testing	The entire software program
Acceptance testing	The entire software program with live data for production readiness

Ideally, these phases should be completed sequentially However, when development work is done in parallel, module coding, unit testing, integration testing are commonly integrated, followed by system testing and acceptance testing

During the process of testing, apply a combination of verification and validation techniques Unit and integration testing are primarily used for program verification These two forms of testing comprise structural testing, which is used to uncover errors injected during program coding System and acceptance testing are used for program validation, and these two forms of testing comprise functional testing, which is used to uncover errors that occurred when implementing requirements or design specifications The following sections will cover some useful testing techniques and scenarios for Year2000 testing

Structural Testing Techniques

Structural testing ensures sufficient testing of a function's implementation and helps determine that all structures of the system are integrated to form a cohesive unit

Operations Testing

Apply operations testing to determine whether the system is ready for normal system (production) operations In contrast, recovery processing (discussed below) is intended for abnormal system operations Considering the potential scope and magnitude of your Year2000 transition, every aspect of the normal operation might be impacted to some extent as you revise programs and/or data for Year2000 readiness. Operations testing ensures that, prior to production, your IS staff can properly administer the applications using the new support mechanisms, documentation, procedures, and training as you complete your Year2000 transition.

Stress Testing

Apply stress testing to determine if the system can function when transaction volumes are larger than normally expected The typical areas that are stressed include disk space, transaction speeds, output generation, computer capacity, and interaction with people. When testing Year2000 changes, it is essential to verify that the existing resources can handle the normal and abnormal volumes of transactions after the restructuring of the code and the possible expansion of the data fields For example, apply stress tests to determine:

© Copyright IBM Corp 1995, 1996

- if existing CPU capacity is sufficient to meet expected user turnaround time when Solution #3 (refer to "Solution #3. A 2-Digit Encoding/Compression Scheme" on page 4-6) is applied and uses more CPU cycles and processing time for code conversion.
- if existing disk capacity is sufficient to accommodate the additional disk space and provide acceptable disk access time when Solution #1 (refer to "Solution #1 Conversion to Full 4-Digit-Year Format" on page 4-1) is applied and expands the year data field from two to four digits.

Recovery Testing

Apply recovery testing to ensure that the system can restart processing after losing system integrity. This is essential for systems in which the continuity of operation is critical to end users. Recovery processing normally involves the ability to go back to the last checkpoint, then reprocess up to the point of failure. The success of the recovery depends heavily on complete backup data and checkpointing. Any data integrity or unresolved exposures that lead to inconsistent data or code after you have implemented appropriate Year2000 solutions will affect the completeness of backup data. On the other hand, checkpointing is very time oriented and sensitive. Any mis-handling of the time-related data might invalidate system checkpointing. The recovery testing is thus critical in a Year2000 testing environment. It can also involve manual functions (such as hardware or operating system failure), loss of data base integrity, operator error, or loss of input capability. Recovery testing should include all aspects of the recovery processing.

Functional Testing Techniques

Functional testing is designed to ensure that the system and end-user requirements and specifications are achieved. Functional testing focuses on the results of processing rather than how processing is implemented. To accomplish this, create test cases to evaluate the functional correctness of the system and programs. Functional testing techniques include:

Requirements Testing

Apply requirements testing to verify that the system performs its function correctly and that it remains functional over a continuous period of time. Functional checklists such as user requirements, design specifications, compliance of organization's polices and procedures are used to create test cases to ensure that these requirements are still satisfied following your Year2000 transition. Note that if the Year2000 solutions are merely restructuring code and reformatting data without major redesign of the applications or systems, most requirements testing can be covered by another method, regression testing.

Regression Testing

Apply regression testing to ensure that all aspects of a system remain functionally correct after changes have been made to a program in the system. Because the potential exists for a tremendous amount of data and programs to be involved in your Year2000 transition, any change to an existing program in the system can have a snowballing or cascading effect on other areas in the system. A change that introduces new data or parameters, or an incorrectly implemented change can cause a problem in previously tested parts of the system, simply because of the way data can be shared between software entities.

Regardless of how an error was introduced or propagated, regression testing needs to be conducted to retest even unchanged parts or programs of the system. Normally, tests that have been previously run are reused to ensure that the same results are achieved. In most cases, regression testing is automated because the test cases and the results are already known.

Error Handling Testing

A normal error-handling cycle is an iterative process that either prevents errors from occurring, or recognizes and corrects errors that have occurred. Error-handling testing is necessary to determine the ability of the system to properly process incorrect transactions that can be reasonably expected as types of error conditions. For example, programs that accept only 4-digit-year-data-entry format need to provide error messages for data entry in 2-digit-year format, and vice versa for programs that accept only 2-digit-year-data-entry format. When changing from 2-digit-year format to 4-digit-year format, you need to apply error-handling testing to verify the appropriate error-handling functions.

Manual Support Testing

Apply manual support testing to evaluate the adequacy of the processes used by people (end users) who must handle the new data generated from the automated applications with Year2000 support. Types of data from these applications include data entry and report generation. Any new data format should be easy to understand and not ambiguous. This method includes testing the interfaces (for example screens, procedures, operation manuals, and online HELP panels) between end users and the application program. End users should be trained and use procedures provided by the system personnel. Testing should be conducted without any other assistance.

Intersystem Testing

Applications are frequently connected with other applications to provide a higher or deeper level of functionality. Data may be shared between applications or systems. Multiple applications or systems may be involved in such an environment. This is the typical environment for Year2000 projects. Intersystem testing is required to ensure that the connection functions properly between the applications. This test determines that the proper parameters and data are correctly passed between applications, and proper coordination and timing of each function exists between applications.

Parallel Testing

Parallel testing is used to determine whether the processing and results of a new version of an application are consistent with the processing and results of the previous version of the application. It should be applied when the old and new versions of the application are similar. For Year2000 solutions without any major function redesign, this is the ideal technique. Parallel testing requires that the same input data be run through the two versions of the application. However, if the new application changes data formats, such as reformatting the year-date notation to 4-digit format, you must modify test input data before testing.

The efficiency and effectiveness of parallel processing is highly dependent on the degree of difficulty encountered in verifying output results and preparing common input It may be difficult to automatically verify the results of processing by comparing the results on a tape or disk file Some automated test tools or customized solutions can be used to prepare input and verify output more quickly

How to Change Date and Time for Testing

By nature, Year2000 exposures are time-sensitive and time-driven Basic Year2000 testing requires that you set the system date and time to a point where Year2000 exposures can be detected, then removed

Attention: Be cautious before resetting the system timer Some system resources and functions are time-sensitive and may be activated or de-activated when you reset the system clock. Such effects can occur when you either set the system clock forward or backward Without careful planning you could cause the loss of these system resources and/or functions, some of which might prove very difficult and time-consuming to recover Ensure that you do not contaminate your production system or production data bases when running various test scenarios Consider

- Providing a separate test system (such as a separate LPAR for large operating systems) and storage device
- Providing a separate set of test data
- Initially turning off RACF or other date-dependent functions (To maintain RACF security, consider implementing a RACF installation exit to disable expiration date processing)

The most vulnerable resources and functions subject to expiration include

- user IDs
- passwords
- data files and databases
- authorization/protection
- licences/services
- network access
- automation functions (as well as unexpected activation)
- hierarchical storage management

For example, if your system is set up to scratch all files that are 1 year old, all files will be scratched when the system clock is changed from any date prior to 1999/01/01 to 2000/01/01

Two ways you can change the system date and time include

1 Use a system console command to set the system timer For example,
   - **On MVS:** reply to message IEA888A [GMT DATE=...,CLOCK=] LOCAL DATE=...,CLOCK= REPLY U, OR GMT/LOCAL TIME in a test LPAR (Do not set the time of day (TOD) using the Sysplex Timer.)
   - **On VM/ESA:** systems will IPL correctly with a year of 2000 or later For VM/ESA Version 1 Releases 2 1 and 2 2, you must apply APAR VM57927 Beginning with VM/ESA Version 2 Release 1, this capability is included in the base VM system

In order to change the date of your VM/ESA system, reply 'Yes' to the `Change TOD clock (Yes|No)` prompt during IPL, and change the date and time to the desired date and time. To specify a year of 2000, simply enter `00` as the year (`01` for 2001, `02` for 2002, and so forth).

When the system date is changed, issue a SHUTDOWN REIPL command immediately after completion of the system IPL as indicated in message HCPITM1161I. This is required because by the time VM issues the `Change TOD clock (Yes|No)` prompt, components of the control program (CP) have already developed sensitivities to the actual value of the TOD (time of day) clock. These sensitivities might cause irregular scheduling and dispatching behavior if the value of the TOD clock is changed by more than several minutes.

- **On TPF:** use the ZATIM (alter time) functional message to change the system time-of-day (TOD) clock, change the subsystem local standard time (LST) clock, and synchronize the TOD clock to a Sysplex Timer. Do so as follows:
  - Issue the ZDTIM functional message to determine the time base before you issue this functional message.
  - If you issue this functional message when the TPF system is above 1052 state, you must cycle the TPF system to 1052 state to complete the time adjustments.
  - You can issue this functional message with the TOD parameter only in 1052 state.
  - In a loosely coupled system, all other active processors must be in 1052 state in order to change the TOD clock without the BP option.
  - The ZATIM functional message does not adjust time-initiated functions (that is, functions that were started by using the CRETC macro).
- **On VSE:** use command `SET DATE=, CLOCK=, ZONE=` to initialize the system timer during IPL time. The DATE parameter is specified in the format mm/dd/yy, whereby yy is interpreted as 20yy, when yy < 50, and as 19yy otherwise. However, the DATE and CLOCK parameters are effective only on a native VSE system. Under VM, VSE does not control the clock and IPL for a date after 1999 depends on VM support, that is not yet available with current VM/ESA releases.
- **On AS/400:** use the CL command (change system value (CHGSYSVAL)) with the QDATE parameter. For example, CHGSYSVAL QDATE('101300') changes the system date to October 13, 2000 on a system using MMDDYY date formats (QDATFMT). To make sure all jobs on the AS/400 are using this new date, you should then power down the AS/400 (PWRDWNSYS) and conduct your testing after the subsequent IPL.
- **On PC:** use a CMOS setup utility or execute the DATE command in DOS Version 3 Release 3 or later.

2. Intercept the call to date and time routines or system timer services. Change the date and time value returned from the routines/services to a specific value that will cause exposures. For example,
   - On MVS, trap the MVS Time macro with LINKAGE=SVC (SVC 11), Time macro with LINKAGE=SYSTEM (SYSPLEX timing services) and STCKSYNC macro

Note:

- There are tools in the marketplace that provide the function of time simulation. Refer to "Solution Developer Tools" on page 7-71 for a list of Solution Developer tools For any time simulation tool, the change of time should be on application-level programs and should not affect the system functions and operations The tool should allow the users to specify the scope of the applications with time change Once the scope is specified, the change of time should be within that pre-defined scope and transparent to others applications.
- When you request full coverage of time references, you must ensure that all forms of time references are intercepted For example, some date/time service routines use hardware instructions to reference time, which are normally not intercepted by most tools

Basic Testing Scenarios

The scenarios for Year2000 testing depend heavily on the system environment and applications Some basic Year2000 testing scenarios that are common for most installations are suggested here

- Set the clock to test process cycles and automatic functions that are activated on a regular basis. These scenarios can be used to identify Year2000 exposures that need to be fixed as well as to validate programs after applying Year2000 solutions
  - Daily
  - Weekly
  - Semi-monthly
  - Monthly
  - Bi-monthly
  - Quarterly
  - Semi-annually
  - Annual
  - Automatic archiving
  - Automatic restart/restore
  - On demand
- Test the setting and display of special dates, including
  - 1900/2/29 - should fail - the year 1900 is not a leap year
  - 1996/2/29 - should succeed - the year 1996 is a leap year
  - 2000/2/29 - should succeed - the year 2000 is a leap year
  - 00/01/01 - should display an unambiguous 4-digit-year date, the value of which depends on the application. For example, 1900/01/01, 2000/01/01, and so on
  - 1999/12/31 - should be able to distinguish between a regular end-of-year 1999 date and a special meaning date For example, a never-expiring date indicator
- Test the processing of time-sensitive data with different combinations of data and time
  1. Use the current system clock and then test data with dates:
     - before 2000/01/01
     - after 2000/01/01

2. Set system clock before the year 2000, for example 1999/12/31, and then test data with dates
   - before 2000/01/01
   - after 2000/01/01
3. Set the system clock after 2000/01/01 and then test data with dates
   - before 2000/01/01
   - after 2000/01/01

Basic Scenarios to Test Your PC System Clock

Some older models of the PC may not have the capability to set or roll over the system clock beyond the year 2000 because the Basic Input/Output System (BIOS) is unaware of the century digits. Refer to "IBM Personal Computer (PCs) - Hardware Timer Setting" on page A-24 for PC-specific, internal clock setting information. Some suggested scenarios for testing for year2000-readiness of your PC system clock follow

- Test if the system clock can be set beyond the year 2000
  1. Set the system clock to 2000/01/01, 00 01 00
  2. Check the date
  3. If the date is set correctly, power off, power on, and then re-check the date
- Test the system clock automatic update function
  1. Test the system clock automatic update function when the power is on
     a. Set the system clock to 1999/12/31, 23 58 00
     b. Keep power on
     c. Wait until the clock reaches the year 2000
     d. Check the date
     e. If it is set correctly power off, and re-check the date
  2. Test the system clock automatic update function when the power is off
     a. Set the system clock to 1999/12/31, 23 58 00
     b. Power off
     c. Wait until the clock reaches the year 2000
     d. Power on
     e. Check the date
- Test the time update by the operating system
  1. Test the time update after suspension of a time-sensitive program
     a. Set the system clock to 1999/12/31, 23 58 00
     b. Suspend a time-display program without a 'wake-up' timer
     c. Keep the power on
     d. Wait until the clock reaches the year 2000
     e. Resume time-display program and check the date
  2. Test time update after suspension and 'wake-up' of time-sensitive program
     a. Set system clock to 1999/12/31, 23 58 00
     b. Suspend a time-display program with the 'wake up' timer set at 2000/01/01, 00 01 00
     c. Keep the power on
     d. Wait until the time display program 'wakes up'
     e. Check the date

Chapter 6. Migration Consideration for Year2000 Transition

To accomplish a successful migration and eliminate your current Year2000 exposures, you will need to follow a well-architected migration plan and prepare to execute that plan prior to actually performing the migration This section provides an outline that you can use as a checklist of those steps that will help you plan, prepare, and execute your migration Note that some steps may not be necessary for your specific environment.

Plan for Migration

1 Plan for Migration
  - Determine the sequence of steps needed for migration
  - Review the migration procedures with your system administration staff and your end-user community
  - Determine the resources/time required for migration
  - Assign individuals/organizations to each migration step
  - Document the migration sequence and responsibilities
  - Develop a schedule for migration of the new system to reach production mode

2 Examine all changed data that use new/changed date format
  - Determine the source of the new data in the existing systems
  - Determine the data that can be converted automatically
  - Determine the data that must be converted manually

3 Design bridges/interfaces among packages and reusable modules to maintain compatibility, if needed
  - Design bridges/interfaces to application packages, if needed
  - Design bridges/interfaces to reusable application systems, if needed
  - Design bridges/interfaces to old systems that will coexist with the new systems, if needed
  - Design tests for the verification and validation of these bridge facilities, if needed

4 Design procedures for manual data conversion
  - Update documents/procedures that will be used for manual data entry
  - Determine checking mechanism for the accuracy and completeness of manually entered data
  - Design the new screens with new date format for manual entry of new data and review with your end-user community
  - Design/update the software to load the manually prepared data into the new system
  - Run a rehearsal of the manual data entry and estimate the impact of the new data format on data entry time

5 Design procedures for automated data conversion

© Copyright IBM Corp 1995, 1996

- Design new software or use automated tools for automated data conversion
- Determine a checking mechanism for the accuracy and completeness of automatically converted data
- Design recovery procedures for conversion of data errors caused by missing data
- Estimate the resources and time for automated data conversion

6 Develop the data conversion systems

- Develop subsystems to convert existing data
- Develop subsystems for the entry of new data
- Develop bridges/interfaces to old systems which will remain in production
- Develop bridges/interfaces to application packages and reusable modules
- Verify and validate the accuracy of the data conversion systems

7 Plan the hardware installation for new system, if needed

8 Plan for final system testing

- Determine the testing strategy
- Develop the detailed test plan and schedule
- Determine the types of tests to be conducted on the new system
- Plan the testing environment
  - Design the migration tests for the systems and applications
  - Determine what testing software or tool(s) will be used for each type of testing
  - Determine what testing libraries will be used for each specific set of programs and data
  - Install needed testing software For example, test data generator, test utilities, debugging utilities, and so on
  - Build test libraries, and test data
  - Coordinate the testing with your development team and your system administration staff

9 Documentation and training

- Update your 'corporate standard guideline'
- Update your technical documentation For example, development guidelines and testing handbooks
- Update the production procedures
- Update the user documentation
  - Update the on-line documentation, including HELP screens, on-line manuals, computer-aided training, and so on
  - Test the on-line HELP and training aids with your end users to evaluate the acceptance of the new on-line information

  - Update all hard-copy documentations to reflect changes and review those documentation with your end users
  - Plan and conduct training program to smooth the migration process

Perform Migration

1. Update production procedures, if necessary
2. Install the Year2000-ready production system environment
    - Coordinate with vendors for the hardware installation, if needed
    - Install the hardware of the new production system, if needed
    - Coordinate with system programmers/operators for installation of the Year2000-ready software
    - Install the Year2000-ready system/vendor/application software on the new production system
3. Perform data conversion process
    - Load existing data into the new system's databases
    - Execute the data conversion programs or automated tools for data conversion
    - Load manually-prepared data through data entry
    - Integrate the existing and converted data
    - Test the integrated data to verify data integrity
4. Perform final system/migration testing
    - Plan the sequence in which separately developed subsystems will be tested and verified in reasonable combinations
    - Verify that the portions of the system that have no changes still runs properly as changes are made to other portions of the system
    - Verify that the program handles all its transactions correctly and remain stable for a defined period of time
    - Verify that the system can accept input from, and provide output to, other systems with which it interfaces as interfaces change
    - Verify end-user acceptance of the new system to certify the system as acceptable for production
5. Activate the new system in production
    - Switch the new system to production mode
    - Run the new system in parallel with the old system
    - Phase out the old system as the new system becomes stable
6. Migration review
    - Monitor and evaluate system performance, throughput, and reliability
    - Determine what system tuning is needed based on system status records
    - Track and evaluate user acceptance of the new system
    - Determine and document what system and application function enhancements are needed

- Plan and schedule the system and application function enhancements
- Coordinate system function enhancements with vendors
- Design and develop required in-house application-function enhancements
- Determine when the system and/or application enhancements will be applied
- Apply the enhancements, once available, to the new system

Chapter 7. Tool Categories and Available Tools to Ease Year2000 Changes

With the critical time constraint of the Year2000 challenge, customers require Year2000-ready applications to be available shortly In addition, customers demand Year2000-ready applications of high quality and reasonable cost It would not be possible to achieve this without the use of powerful and productive tools There are a variety of tools available from Solution Developers that can help you confront the software challenge of the 1999 to 2000 date change Refer to "Solution Developer Tools" on page 7-71 for a partial list of Solution Developer tools.

To be effective and efficient in providing the ability to rapidly change programs and data to properly handle the year 2000, the tools must have certain characteristics Some important tool characteristics and tool types that are necessary to make these changes are summarized in this chapter This chapter also provides lists of tool products being marketed by Solution Developers and IBM as general information and as a reference to obtain further information

Tool Characteristics

The tools should provide necessary features such as

- interactive environment
- batch processing capability
- graphic user interface
- ease of use
- rapid prototyping
- speedy and easy editing/updating
- stepwise refinement
- backward recovery

In addition, consider both the software ***development*** and ***deployment/target*** environment of the tools such as

- Platform
  - Host only
  - Workstation only
  - PC only
  - Client/Server-based
  - Cooperative processing with host
  - LAN based
- Prerequisite hardware (both minimum required and recommended for PC specification, if necessary) such as:
  - memory/storage required
  - DASD needed for tool installation
  - DASD needed for tool usage
- Prerequisite software that is required and supported such as:
  - Host operating system(s) - for example MVS, VM, VSE, OS/400, Windows NT, OpenVMS, AIX

© Copyright IBM Corp 1995, 1996

  - PC/Workstation operating system(s) - for example DOS, Windows, OS/2, Unix, MacOS
  - Network operating system(s) - for example Netware, Banyan, Windows NT
  - Communication protocols - for example TCP/IP, SNA, IPX, NetBios
- Languages supported or generated such as COBOL, C, C++, PL/I, FORTRAN, RPG, PASCAL, Smalltalk, Assembler
- Client/server models supported such as:
  - Transactional - for example CICS, Encina, Tuxedo, Remote Procedure Call (RPC)
  - Conversational - for example APPC/CPI-C, NetBios, IPX, TCP/IP, SNA
  - Database server - for example DB2 family, Sybase, Oracle, EDA/SQL, SQL Server

Once you run the appropriate tools against your operating system to reformat Year2000 exposures, be certain that the 'newly created' system maintains its ability to:

- Achieve good machine performance
- Process a reasonably large number of users
- Process reasonably large databases
- Process high-traffic volumes
- Provide networking access
- Provide recovery from failures
- Provide security
- Provide audibility or accounting
- Provide ease of maintenance

Tool Categories

The following sections highlight some useful tool types for handling the year 2000 challenge. A brief description is provided for each of the types. Most of these tools are used in normal prototyping and application development. Tool types can be categorized as follows:

Impact Analysis

To analyze the impact to your programs, you can use tools to:

- ***analyze complexity***

  Determines the complexity of a software design or code using a metric such as 'fan-in/fan-out', degree of nesting, or other characteristics. These tools provide complexity analysis to allow you to estimate the effort required to change the date/time-related items in your source code.

- ***analyze impact***

  Analyzes the program modules and related data to determine what is impacted and related. These tools are very time-efficient but do not always guarantee the accuracy of their analysis because they tend to over-estimate what is affected. When an impact analysis tool indicates that some data is affected, it does not always mean that the data will need to be changed (such as to reformat the date-related data and programs for Year2000 readiness).

- ***analyze metrics***

  Collects, analyzes, and reports the results of metrics quantification and analysis activities. These tools can analyze and predict how much work, in quantification, will be needed to reformat the date related data and programs for the year 2000, based on a metrics or cost model. You must validate the accuracy of the metrics you intend to use, that is, its predictions against actual human performance in real-life situations.

- ***analyze database***

  Investigates the structure and flow within a database to observe the characteristics of the database and determine if certain measurements/requirements can be realized. For example, analyze the year fields of the databases for any use and cross reference of 2-digit years.

Project Management

To help manage the project, you can use tools to:

- ***inventory software***

  Determines all code, JCL, databases, and other programs that constitute your system to provide a complete list for impact analysis. The list can be further divided into lists of sub-systems when partitioning and prioritization of the project is necessary.

- ***track changes***

  Tracks and logs all requests for code and/or data changes. Requests are tracked through completion or resolution. Any inconsistent/missing changes of data or programs due to date format changes will then be minimized.

Program Level Analysis

To analyze programs and data, at a program level, you can use tools to:

- ***analyze data flow***

  Shows the flow of data among modules in procedures or programs. Determines if a data-flow diagram is complete, consistent, and adheres to those rules established that govern flow. This provides both high- and low-level views of data flow within the system, and can be used to verify completeness of the program and data changes.

- ***diagram logic structure***

  Diagrams how program modules call sub-modules, and what data and control information these program modules share. These tools display multiple program views.

- ***diagram data structure***

  Diagrams the representation of appropriate parts of a data model as the structure is used by a database management systems structure and relational structures.

- ***diagram relationships***

  Illustrates multiple relationships of a program module or data element at the same time. This is useful to understand how data is shared among the programs that have access to it.

- ***diagram decomposition***

Allows a high-level overview specification, for a design or data model, to be successfully decomposed into smaller entities for further observation and analysis It facilitates the partitioning of a project that is too large to tackle all at once, such as the Year2000

- ***slice programs***

  Allows you to view all the code affecting a given variable or statement Forward slicing starts with a name or statement, and indicates what that name or statement affects Backward slicing starts with a name or statement, and indicates all the parts of the program that *could* affect it

- ***analyze logic***

  Inspects the use of control logic within a program, determines if it is proper, and mechanizes the specified design. It is useful for verification and validation of the correct manipulation of time when windowing techniques are used

Code Editing and Restructuring

To help edit and restructure code for your programs, you can use tools to

- ***power browse***

  Allows you to scan and inspect code Scanning can be switched between program (data) structure charts and code These tools are more powerful than regular text editors They can provide, for example, syntax checking, sophisticated capability to find data or information, or the ability to edit multiple programs. These types of tools can also include reverse engineering tools or maintenance workbench tools

- ***find dates***

  Locates date-oriented data, variables, declarations, comments, or other information in code for investigation of potential reformatting

- ***comparison***

  Compares two software programs, files or data sets to identify commonalities and/or differences It is extremely useful for verification of program changes when date reformatting is done by simple field expansion

- ***cross reference***

  Lists where variables, procedures, or other items are located in the code These tools speed up browsing and provide a limited form of impact analysis

- ***expand fields***

  Automatically expands 2-digit-year fields into 4-digit-year fields It saves editing time tremendously and provides complete coverage of field expansion

- ***analyze interfaces***

  Determines if a range of variables in the programming interfaces is correct, as the variables are referenced across the reference boundaries. It can designate 4-digit-year format as a standard interface and enforce the standard in the programming interfaces

- ***analyze standard/consistency***

Determines whether prescribed development standards have been followed Identifies inconsistency in conventions used in requirements designs or programs These tools can help you introduce standard or more uniform names to date-oriented fields or keywords, and improve the consistency and accuracy of data. These tools enforce consistent indentation and alignment

- ***trace requirements***

  Traces how the requirements are realized in the design and code

- ***modularize code***

  Generates modular code and top-down control flow Reduces the scope of complex programs by creating separate modules Identifies routines that are frequently referenced or changed, for example date/time services routines and creates re-usable code libraries

- ***standard date subroutine***

  Creates reusable program modules that have correctly implemented date handling These date subroutines can replace individually developed date subroutines to standardize the use of a date routine These tools also reduce the chance of error and the cost of development, maintenance and testing

Code Generation

To generate code for your programs, you can use tools to

- ***generate database code***

  Generates database code directly from the data structure diagram

- ***paint screen***

  Generates code for the screens of a computer-user dialog or data entry when the screens need update for reformatting of date

- ***generate dialog***

  Generates dialogs that conform to specified standards, for example, 4-digit-year input/output standard in any dialog

- ***generate reports***

  Generates code for the structure and layout of a report, along with calculations of derived fields in the report

- ***generate code***

  Generates executable code from high-level specifications

Automate Testing

To automate testing for your programs, you can use tools to

- ***simulate***

  Represents certain features or functions of the behavior of a physical or abstract system One example of such a tool is a clock simulator that can change your system clock, while being transparent to your programs Tools such as these provide an easy way to quickly expose your programs to Year2000 scenarios

- ***analyze tests***

Determines the test case coverage on a set of programs being tested (whether a segment of code had been tested by other testing)

- ***generate test data***

  Generates test data directly from a specification and facilitates a sequence of testing steps

- ***test data libraries***

  Organizes test data for use These libraries are most useful in regression tests

- ***test drivers***

  Automates testing by triggering test cases during testing. These tools often provide test input, execute the test cases, compare actual test output with expected results, and report test results

IBM Tools for MVS

The IBM COBOL Family for MVS & VM

For application code written in COBOL for S/370 and S/390 (for MVS and VM) platforms, IBM has developed COBOL compilers These products target the host application development environment.

IBM's COBOL mainframe products for S/370 and S/390 are

- **COBOL for MVS & VM** (new name for IBM COBOL/370) — the compiler
- **Language Environment for MVS & VM** (new name for Language Environment/370) — the run-time library
- **CoOperative Development Environment** (CODE/370) — edit/compile/debug tool

Features

This COBOL compiler, COBOL for MVS & VM, is available today and provides full 4-digit year support with features including

- Intrinsic functions
- Sliding century window

COBOL for MVS & VM provides full Year2000 support It provides ANSI COBOL Standard Intrinsic Functions which give full date manipulation capability with 4-digit-year support Language Environment for MVS & VM provides additional date manipulation with the Language Environment Callable Services COBOL for MVS & VM will return a 4-digit year when the COBOL application program queries the system for the current date OS/VS COBOL and VS COBOL II return the system current date with a 2-digit year

Companies need productive application development environments A language compiler is only a portion of what application programmers need today to develop and maintain code Tools that generate statistics from code; modularize, migrate, or restructure code; or search class libraries, are becoming more and more important.

These tools assist in migrating your existing COBOL applications to COBOL for MVS & VM

- **CCCA** — converts source code
- **COBOL Structuring Facility** — analysis, reporting, restructuring
- **CICS Application Migration Aid** — converts CICS source
- **COBOL Report Writer Precompiler** — supports Report Writer code
- **EDGE Portfolio Analyzer** — load inventory tool
- **WITT** Product Family — testing
- **ReDiscovery** Product Family — cataloging of parts

IBM COBOL for MVS & VM

IBM COBOL for MVS & VM (program product # 5688-197) is the high-performance compiler with a supporting runtime environment (Language Environment for MVS & VM (program product # 5688-198)), which facilitates multiple-language interaction This is different from the single packaging of compiler and runtime environment for OS/VS COBOL and VS COBOL II. An additional component, CoOperative Development Environment/370 (CODE/370) (program product # 5688-194) provides an integrated workstation development environment for developing host applications It includes an editor, compiler invocation facility, and a debug tool for the host

Applications written using COBOL for MVS & VM can interface with a variety of IBM products, such as SQL/DB, DB2, CICS, GDDM, ISPF, and Data Window Services available on MVS/ESA The consistent interlanguage communication support, common protocols, and suite of callable services provided by Language Environment for MVS & VM is designed to allow easier access to in-house applications or vendor packages written in COBOL for MVS & VM

IBM COBOL for MVS & VM Release 2 brings object-oriented programming to the MVS COBOL programmer Object-oriented extensions to the COBOL language syntax are provided for MVS SOM is the core OO architecture for SOMobjects for MVS and is IBM's strategic architecture for building and manipulating class libraries Enabling to SOM on MVS allows IBM COBOL programmers to develop class libraries using native COBOL language with object-oriented extensions The COBOL SOM objects permit COBOL object-oriented applications to access SOM objects implemented in other languages, in addition to full interoperability with existing COBOL applications and data.

IBM COBOL for MVS & VM provides:

- Intrinsic functions, which reduce the need for extensive algorithms
- Programmer access to all of the elements in a table at once, reducing the need for explicit loops
- Consistent interlanguage communications, common services, and common functions, which helps extend the useful life of existing applications
- Improved dynamic calls
- Support for Year2000
- Capabilities to help application programmers incrementally enhance applications
- Help in maintaining and enhancing the investment in existing programmer skills.

The following new features are included in IBM COBOL for MVS & VM Release 2

- Object-oriented language extensions — allowing developers to create mission-critical business applications that run only on the host or as part of a client/server application These extensions are based on a subset of the evolving ANSI OO COBOL Standard
- Support for the direct creation of SOM objects on the host via COBOL language syntax
- Optional SOM Interface Definition Language (IDL) generation
- Access to existing SOM-based class libraries
- Improved interoperability with C and C++
- Source-level compatibility with IBM VisualAge for COBOL for OS/2 and IBM COBOL Set for AIX
- Performance enhancements

IBM COBOL for MVS & VM Release 2 is compatible with VS COBOL II and IBM COBOL/370 Release 1. For SOM applications, customers must also order SOMobjects for MVS (program product # 5696-822).

IBM COBOL for MVS & VM provides facilities to acquire and integrate packaged software, consistent with the vendor's terms, into existing applications irrespective of the language used, use existing code in new applications (code reuse) regardless of the source code language used, and invoke functionality between applications with improved interlanguage communication (ILC)

IBM COBOL for MVS & VM includes VS COBOL II language features, enhanced compiler options, virtual storage constraint relief, structured programming language for improved programmer productivity, enhanced double-byte character set (DBCS) support, streamlined system interfaces, expanded code optimization, flexible run-time options, and support for the VSE/ESA 31-bit addressing feature.

With IBM COBOL for MVS & VM, the "RES multitasking restriction" is lifted Thus, independent COBOL applications can be executed under different tasks in the same MVS address space In particular, COBOL applications can be executed from both halves of an ISPF split screen.

Most applications were coded with 2-digit-year data. COBOL for MVS & VM provides intrinsic functions not available in our earlier COBOL compilers Calendar functions include

- CURRENT-DATE
- DATE-OF-INTEGER
- DAY-OF-INTEGER
- INTEGER-OF-DATE
- INTEGER-OF-DAY
- WHEN-COMPILED

Example of obtaining the 4-digit year with the COBOL Intrinsic Functions

- DATE-OF-INTEGER gives YYYYMMDD
- DAY-OF-INTEGER gives YYYYDDD

IBM Language Environment for MVS & VM

IBM Language Environment for MVS & VM (program product # 5688-198) is IBM's common runtime environment for enterprise applications written in COBOL, PL/I, C, and FORTRAN Language Environment for MVS & VM is designed to provide defined calling conventions, enhanced interlanguage communication, callable services, language-specific services (for example, common facilities messages), common math functions, application utilities, system services, and subsystem support for Customer Information Control System (CICS) and Information Management System (IMS).

With Language Environment for MVS & VM, application programmers can extend and integrate their applications and packages, as well as reuse code with greater flexibility, due to interlanguage communication and native language restrictions Application packages may be extended with the language of choice. Language Environment for MVS & VM enables existing applications to function as before with little, if any, changes required, thus helping preserve company investment in those applications. Language Environment for MVS & VM condition handler permits programs to handle errors in a predictable, logical manner without highly-specialized routines.

Language Environment for MVS & VM replaces the existing language-specific run-time libraries and provides a common runtime environment for all languages that conform to the Language Environment architecture. In a single product, Language Environment for MVS & VM combines essential run-time services, such as routines for message handling, condition handling, and storage management. All these services are available through a set of interfaces that are consistent across programming languages With Language Environment for MVS & VM, application programmers can use one runtime environment for their applications, regardless of the programming language or system resource needs

Today, IBM Language Environment for MVS & VM is the run-time library for the Language Environment-enabled compilers IBM COBOL for MVS & VM, C/C++ for MVS/ESA, and PL/I MVS & VM. IBM realizes the importance our customers place on complete, open, high-quality solutions. Therefore, IBM has established an *IBM Language Environment Partner Program* at the IBM Santa Teresa Laboratory This program encourages and assists tool and application package vendors to support and take advantage of Language Environment services Companies with existing 3GL applications will benefit from the wider choice of tools and application packages

Language Environment for MVS & VM provides a number of advantages over other packages: its capabilities include

- Ability to parse dates in an infinite number of formats by using a picture string feature as a parsing guide
- Provide NLS support by using a built-in table of defaults based on a country code
- A sliding window feature

  IBM Language Environment for MVS & VM provides a valuable short term solution with the century window feature If you are unable to change all of your applications and data at the same time, the century window feature allows 2-digit years to be interpreted in a 100-year window Any 100-year window can be selected You pass a 2-digit year to Language Environment and Language Environment returns a 4-digit year based on the 100-year window

  The advantage to the century window is that you need to change only the application code and not the databases or files with 2-digit years This allows you to change the application programs one at a time or groups at a time without effecting your data files Note, this only works for dates that range less than 100 years For example, dates of birth may not be appropriate for this solution, a person born in '94' could be over 100 years of age

  The disadvantage to the century window is that it doesn't last forever If your application has a long life expectancy, you many need to go back and replace the century window with full 4-digit year support Some of your applications will be replaced prior to this replacement

IBM CoOperative Development Environment/370 (CODE/370)

IBM CoOperative Development Environment/370 (CODE/370) (program product # 5688-194) — the common editor, compiler, debug tool — provides a cooperative environment, allowing application programmers to more productively develop and maintain host IBM 3GL applications from the workstation. CODE/370 provides a consistent, graphical user interface across different platforms and languages, a language-sensitive editor, language-sensitive help, a compiler invocation facility, and an interactive debug tool. CODE/370 combines the richness of the S/370 or S/390 subsystem environments and the power of IBM Language Environment for MVS & VM to provide a host Debug Tool which allows programmers to find bugs, fix bugs, and test applications The Debug Tool is available either as a stand-alone 3270 host debug tool (for programming shops where workstations are not available) or with an optional graphical workstation user interface

CODE/370's cooperative environment allows application programmers to perform host programming tasks, such as compiling and debugging, from a workstation Through cooperative processing, users perform functions locally on the workstation while interactively accessing the programs, data, and compilers residing on a host system. The optional workstation interface combines the Edit and Compile / Link functions together with the Debug Tool graphical user interface (GUI).

The powerful workstation-based editor integrates a rich set of functions that will speed up your application-development activities The editor works with any type of source. The language-sensitive features help optimize coding efficiency in COBOL, C, PL/I, REXX, and JCL. Source can be stored in an MVS data set, a VM file, or an OS/2 file As you edit source in any of these formats, CODE/370 maintains the sequence numbers and date stamps.

This method of workstation/host tool integration offers the best use of the two environments the S/370 subsystem environments' "live" debug capabilities and the workstation GUI's easy editing capabilities The integration also allows programming shops to grow at their own pace into developing host applications on workstations

When programmers use the Debug Tool, the debug session is recorded in a log file, permitting edit and replay of a Debug Tool session. This allows the Debug Tool to be used to capture test cases (for future program validation) or to further isolate a problem within an application. This also allows both interactive and batch debugging of a programmer's application

CODE/370 provides

- Support for COBOL/370 Rel 1, COBOL for MVS & VM Rel 2, C/370 and PL/I MVS & VM
- Limited support for VS COBOL II Rel 3 1, 3 2, and 4 0 and OS PL/I Ver 2 Rel 1, 2, and 3
- A 32-bit, user-programmable editor
- Several language-sensitive editing features for REXX and JCL
- Enhanced support for debugging under CICS
  - Pseudo conversational transaction support for COBOL applications
  - Support for the full range of single-terminal Send and Receive messages
- Program Generator, an independent compile/link program that allows compiling from inside and outside the editor
- REXX and JCL programs can be submitted to the host from the Editor window
- An OS/2 desktop tool called WorkFrame/2
- Advanced Program to Program Communication (APPC) protocol support for cooperative sessions between the MVS host and the workstation
- Support for debugging COBOL applications consisting of multiple enclaves and multiple processes
- Debug Tool support for exception handling of COBOL IGZ exceptions
- The ability to perform initial installation of CODE/370's workstation feature from a LAN server

Other Host COBOL Compilers for MVS & VM

IBM has developed three COBOL products for the MVS & VM mainframe: OS/VS COBOL, VS COBOL II, and COBOL for MVS & VM.

OS/VS COBOL was withdrawn from marketing in June of 1992 and withdrawn from service in June of 1994. This section discusses issues involved with migrating from OS/VS COBOL. The two COBOL mainframe products that will continue to be available are COBOL for MVS & VM and VS COBOL II.

Migrating from OS/VS COBOL: As announced in June 1992, OS/VS COBOL products are discontinued, effective June 1994. A service extension for OS/VS COBOL is available in certain geographic locations.

Because OS/VS COBOL (program product # 5740-CB1) has been discontinued we are encouraging companies to upgrade their COBOL technology to COBOL for MVS & VM (compiler) with Language Environment for MVS & VM (run-time library). This is especially important for your Year2000 solution providing 4-digit year support. Depending on which product is currently being used and how fast a company is willing to migrate, there are three possible migration paths that you can follow.

Migration Paths: The paths are

1. OS/VS COBOL------------------→COBOL for MVS & VM

   Companies with OS/VS COBOL are encouraged to migrate directly to IBM COBOL for MVS & VM (compiler) and IBM Language Environment for MVS & VM (run-time library) if the IBM Language Environment for MVS & VM prerequisites are satisfied. See the IBM Language Environment for MVS & VM Licensed Program Specification for prerequisite information.

2. OS/VS COBOL----→VS COBOL II----→COBOL for MVS & VM

   Companies with OS/VS COBOL who do not yet satisfy the IBM Language Environment for MVS & VM prerequisites must migrate to VS COBOL II first. When the IBM Language Environment for MVS & VM prerequisites are satisfied, then a company can move to IBM COBOL for MVS & VM.

Each successive COBOL product contains more capabilities than the previous product.

VS COBOL II: VS COBOL II (program product # 5668-958) builds on the functions of OS/VS COBOL but has a variety of features that give companies many advantages over earlier IBM COBOL products. VS COBOL II includes additional language features, enhanced compiler options, virtual storage constraint relief, structured programming language for improved programmer productivity, enhanced double-byte character set (DBCS) support, streamlined system interfaces, expanded code optimization, flexible run-time options, and support for VSE/ESA's 31-bit addressing feature.

Companies that do not yet satisfy the IBM Language Environment for MVS & VM prerequisites should run VS COBOL II.

Migration Path:

VS COBOL II----→COBOL for MVS & VM

Companies with VS COBOL II can migrate to IBM COBOL for MVS & VM once the IBM Language Environment for MVS & VM prerequisites are satisfied

Benefits of COBOL Migration:OS/VS COBOL is the ANSI 74 compiler ANSI 85 introduced many significant functions which are provided to customers in either VS COBOL II or COBOL for MVS & VM and its companion product, COBOL for VSE Customers who have migrated to the newer COBOL Standard have additional functionality, increased developer productivity and exploitation of S/390 hardware capabilities Some benefits of upgrading COBOL technology are

- Improved Interlanguage Communication (ILC)
- Condition management features of Language Environment, which bring PL/I-like condition handling to COBOL and C in MVS & VM
- Language Environment callable services, including a date/time service routine that interprets a 2-digit year to a 4-digit year to accommodate the year 2000
- Improved application performance of COBOL for MVS & VM compared to VS COBOL II
- Increased maintenance productivity from restructuring tools such as IBM's COBOL Structuring Facility
- CoOperative Development Environment/370 (CODE/370), which provides increased programmer productivity compared to traditional host development tools
- Object-oriented extensions in COBOL for MVS

For more information on the value of migrating from OS/VS COBOL or VS COBOL II to COBOL for MVS & VM, obtain a copy of *Why Migrate to COBOL/370 and LE/370?*, which is available from your IBM representative (COBMGVAL PACKAGE on MKTTOOLS). This document contains examples of customer benefits of migrating to COBOL for MVS & VM.

COBOL and CICS/VS Command Level Conversion Aid (CCCA)

COBOL and CICS/VS Command Level Conversion Aid (CCCA) (program product # 5785-ABJ) is an effective tool designed to make it easier to convert old COBOL source code and copy modules to the new COBOL Standard CCCA converts OS/VS COBOL, DOS/VS COBOL, and COBOL 74 Standard VS COBOL II (either VS COBOL II Release 1 and 2 or VS COBOL II Release 3 and 4 (CMPR2)) source code to COBOL 85 Standard VS COBOL II Release 3 or 4 (NOCMPR2) or to IBM COBOL for MVS & VM.

In cases where a statement is no longer supported and has no equivalent statement in the target COBOL, CCCA flags the statement. CCCA can be used to convert from OS/VS COBOL to COBOL for MVS & VM, just as it is used to convert from OS/VS COBOL to VS COBOL II. The source file output for compiling under VS COBOL II can also be used for compiling under COBOL for MVS & VM

When converting from OS/VS COBOL, CCCA can be automatically invoked by COBOL/SF, a re-engineering tool. This lets users convert programs before structuring them with COBOL/SF

CCCA is designed to identify and convert source code incompatibility, to reduce tne effort required to convert programs, and to minimize conversion errors The conversion process can be customized by users to meet unique conversion requirements Installation and usage are easy, fast, and straightforward

CCCA key benefits are

- Identification and conversion of source code
- Reduction of the effort required to convert programs
- Minimization of conversion errors
- Enhanced programmer productivity during migration

CCCA provides facilities to

- Convert most syntax differences between OS/VS COBOL, DOS/VS COBOL, or VS COBOL II Release 1 or 2 and the current release of VS COBOL II and COBOL for MVS & VM programs
- Convert EXEC CICS commands
- Remove and/or convert the base locator for linkage (BLL) section mechanism and references
- Eliminate conflicts between user-defined names and words reserved for VS COBOL II
- Convert both source programs and copy modules
- Create conversion management reports
- Produce a statement-by-statement diagnostic listing showing the result of the conversion process for each program
- Change and/or create COBOL conversion modules
- Allow foreground conversion of CICS programs
- Perform conversion from various levels of COBOL into other COBOL levels through an open converter design
- Read from PDSEs, not just PDSs

COBOL Structuring Facility

IBM COBOL Structuring Facility/MVS & VM (COBOL/SF) (program product # 5696-737) reduces the amount of time needed to maintain code by automatically transforming complex unstructured programs into structured programs Improved maintenance productivity frees programmers to focus on creating new applications COBOL/SF promotes application redevelopment by providing a set of complexity metrics for inventory analysis Information provided on program complexity and control flow can promote code reuse and speed up the development of new applications.

COBOL/SF provides a connection to COBOL and CCCA that automates the conversion of COBOL code to a higher-level standard and new COBOL technology COBOL/SF also provides access to VIA/Renaissance (ViaSoft), which does program slicing to extract program logic from COBOL source code and generate self-contained program modules. These three tools can be used together to automatically convert, restructure, and modularize COBOL applications for use in maintenance, code sharing, new development or modularizing any date calculation or processing

COBOL/SF automatically produces a structured COBOL program from unstructured source code In addition, statistical metrics, structure charts to aid in program understanding, and modularization analysis reports are provided

Three steps form the basis for creating program parts that can be used in a client/server application architecture. converting OS/VS COBOL to VS COBOL II or to COBOL for MVS & VM, restructuring, and modularizing to create functional program units If the functional units are wrapped with an object-oriented COBOL wrapper, then developers can use these primitive parts for new OO applications

COBOL Structuring Facility provides

- Analysis of potential problem areas in source code and expert advice on how to manually improve source code quality
- Improved reporting capability and visual program display
- An on-line tutorial
- Comprehensive on-line documentation
- Automatic structuring of COBOL code containing
  - CICS HANDLE commands
  - COBOL for MVS & VM intrinsic functions
  - Language Environment for MVS & VM support
- Cross-reference browser
- Conformance to CUA 1991 standards
- DBCS support consistent with IBM COBOL for MVS & VM enablement

COBOL/SF is more than a one-time restructuring tool It can be used regularly to maintain structured programs for ease of maintenance and program understanding By offering modularization advice, COBOL/SF also proves useful as a redevelopment tool to aid in isolating reusable program functions such as standard date routines. Programming experts recommend restructuring an application whenever 10 to 15% of a program has changed due to a program error or enhancements

CICS Application Migration Aid

The CICS Application Migration Aid (program product # 5695-061) is designed to assist developers in converting CICS applications from the macro-level API to the command-level API Applications written in assembler or COBOL can be used with the migration aid

Old CICS transactions used a macro-level interface However, CICS/ESA V3 3 (which is required when using Language Environment for MVS & VM) does not support the macro level. Command level is a requirement for applications to run on CICS/ESA V3

The CICS Application Migration Aid simplifies and speeds the conversion of COBOL and Assembler language application programs from macro to command level. The tool completely converts simple macros and provides guidance on converting more complex macros

Conversion is key not only to using Language Environment for MVS & VM but also to obtain the benefits of the CICS command-level application interface (API), which is common across the CICS family

COBOL Report Writer Precompiler

The COBOL Report Writer Precompiler (program offering # 5798-DYR) has two functions It can permanently convert Report Writer statements to valid COBOL statements that can be compiled in IBM COBOL for MVS & VM or IBM COBOL for VSE/ESA Or, it can be used to precompile applications containing Report Writer statements so the code will be acceptable to the IBM COBOL for MVS & VM or IBM COBOL for VSE/ESA compiler

When used to precompile, the Precompiler automatically invokes the IBM COBOL compiler—as though Report Writer statements in the source program are being processed by the IBM COBOL for MVS & VM or IBM COBOL for VSE/ESA compiler itself The fact that two separate processes are involved is transparent to users.

EDGE Portfolio Analyzer - COBOL Migration Tool

The EDGE Portfolio Analyzer Version 1 (program product # 5633-009) can significantly reduce the effort necessary to migrate from an earlier version of an IBM host COBOL compiler to the latest level of this product

This migration tool analyzes existing application load libraries and provides data and statistical information about.

- Which release of a compiler and what compiler and linkage editor options are used to produce an existing load module
- Application programs:
  - Containing shared subroutines
  - Requiring re-linking of runtime modules as a result of any arbitrary change
  - Impacted by implementation of a new runtime package
  - Not expected to be affected by the implementation of a new runtime package
  - Containing interlanguage dependencies that may prove obstacles to migration
- Load modules not conforming to the installation's established compiler and linkage editor standards

Customers can save many hours of tedious investigation and move more quickly into the new compiler products

Workstation Interactive Test Tool (WITT)

Workstation Interactive Test Tool (WITT) Product Family currently consists of WITT for OS/2 (program product # 5648-031), WITT for AIX (program product # 5765-222), WITT for Windows (program part # 10H7744), WITT for SUN (program product # 5765-407), and WITT for HP (program product # 5765-406)

COBOL programmers can use the Workstation Interactive Test Tool (WITT) as a GUI test tool that automatically facilitates the creation of reusable unit test, function test, system test, and regression test cases These reusable test cases can be created when the program is newly developed and should be run continuously during the years that the program is maintained Once created, the test cases can be played back interactively or in an unattended batch mode The execution of a set of WITT test cases provides an audit trail of errors that a tester can provide to a programmer to document program failures

WITT automatically records and plays back all keystroke and mouse movements and compares and identifies test inconsistencies between benchmark test cases and test cases run after enhancements and fixes are made to a program WITT also allows scripting in 2/REXX to add program intelligence to the test cases This makes WITT test cases easy to update, maintain, and reuse for future testing.

WITT/OS2 installs on an OS/2 desktop and allows the testing of MVS, VM, IMS, CICS, OS/400, and OS/2 PM and Text (ie, Micro Focus, CICS/OS2) applications WITT/PM installs on an OS/2 desktop and allows the testing of only OS/2 PM and Text applications X/WITT installs on an AIX desktop and allows the testing of AIX X Window applications, as well as remote client applications in X Windows environments on SUN, HP, APOLLO, and MVS WITT/Windows installs on a Windows 3.1 desktop and allows the testing of Windows workstation applications

ReDiscovery

ReDiscovery Product Family consists of ReDiscovery/2 (program product # 5871-AAA 70G3659) and ReDiscovery/MVS (program product # 5655-067)

COBOL developers can reuse code to minimize new coding and enable code sharing between applications This reduces the cost of application development, test, and maintenance while helping to improve code quality ReDiscovery is a tool that facilitates the introduction of a reuse initiative into application development by cataloging parts/modules and improves productivity Reusing information in complex systems that span multiple platforms is extremely challenging. There are a large number of modules to sort through when maintaining an application. Some research, design, and development of new tools or other processes may be duplicated. By using function and data that already exist, programmers can increase the speed of new development and maintenance

ReDiscovery is a productivity enhancement tool that allows programmers to organize and manage information about each file on their computer system Some examples of files are. software parts and components, test cases, data files, source files for documents and manuals, JCL files, etc

By using ReDiscovery to quickly locate parts and the attributes that describe them, programmers can perform some time-consuming and complex tasks quickly and easily. For example.

- **Maintaining Applications** - identifying all modules and data that will require modification due to an error or new requirement in the existing application
- **Language Migration** - identifying all the instructions that need to change across all the modules that are migrating
- **Rightsizing** - identify parts regardless of platform as input to the client-server migration process and downsize, consolidate, or upsize

ReDiscovery facilitates the introduction of a reuse initiative into an application development organization Many applications in an organization may contain reusable functions that could reduce development time Therefore, collecting, organizing, and sharing data about these applications is essential for a successful reuse program.

In addition, ReDiscovery can assist in locating date-related code by searching the ReDiscovery catalogs for keywords such as 'YY' or 'YEAR' or 'YR'.

The IBM PL/I Family for MVS & VM

For application code written in PL/I for S/370 and S/390 (for MVS and VM) platforms, IBM has developed PL/I compilers. These products target the host application development environment

IBM's PL/I mainframe products for S/370 and S/390 are:

- **PL/I for MVS & VM** — the compiler
- **Language Environment for MVS & VM** (new name for Language Environment/370) — the run-time library
- **CoOperative Development Environment** (CODE/370) — edit/compile/debug tool

Features

This PL/I compiler, PL/I for MVS & VM, is available today and provides full 4-digit year support with features including

- Built-in functions
- Sliding century window

PL/I for MVS & VM provides full Year2000 support. It provides Built-in Function which provides 4-digit-year support. Language Environment for MVS & VM provides additional date manipulation with the Language Environment Callable Services. PL/I for MVS & VM will return a 4-digit year when the PL/I application program queries the system for the current date

Companies need productive application development environments. A language compiler is only a portion of what application programmers need today to develop and maintain code. Tools that generate statistics from code modularize, migrate, or restructure code, or search class libraries, are becoming more and more important.

These tools assist in developing PL/I applications with PL/I for MVS & VM

- **EDGE Portfolio Analyzer** — load inventory tool
- **WITT** Product Family — testing

IBM PL/I for MVS & VM

IBM PL/I for MVS & VM (program product # 5688-235) is the high-performance PL/I compiler for the MVS/ESA and VM/ESA environments

PL/I for MVS & VM provides the capability of integrating PL/I applications into IBM Language Environment for MVS & VM. Language Environment for MVS & VM, a common run-time environment, supports PL/I for MVS & VM, COBOL for MVS & VM, and C/C++ for MVS/ESA. Together, Language Environment and its supported languages create a common run-time environment. This integration allows you to take advantage of features from both PL/I and Language Environment. In addition, common function across the supported languages and platforms improves usability as well as programmer productivity

Applications written using PL/I for MVS & VM can interface with a variety of IBM products such as SQL/DB, DB2, CICS, IMS, and Data Window Services available on MVS/ESA. The consistent interlanguage communication support, common protocols, and suite of callable services provided by Language Environment for MVS & VM is designed to allow easier access to in-house applications or vendor packages written in PL/I for MVS & VM

IBM PL/I for MVS & VM together with Language Environment for MVS & VM provide:

- Consistent interlanguage communications, common services, and common functions, which helps extend the useful life of existing applications
- Improved dynamic calls
- Support for year 2000
- Capabilities to help application programmers incrementally enhance applications
- Help in maintaining and enhancing the investment in existing programmer skills

IBM PL/I for MVS & VM provides facilities to acquire and integrate packaged software, consistent with the vendor's terms, into existing applications irrespective of the language used, use existing code in new applications (code reuse) regardless of the source code language used; and invoke functionality between applications with improved interlanguage communication (ILC)

IBM PL/I for MVS & VM includes enhanced compiler options, virtual storage constraint relief, structured programming language for improved programmer productivity, enhanced double-byte character set (DBCS) support, streamlined system interfaces, expanded code optimization, and flexible run-time options

IBM Language Environment for MVS & VM

IBM Language Environment for MVS & VM (program product # 5688-198) is IBM's common runtime environment for enterprise applications written in COBOL, PL/I, C, and FORTRAN. Language Environment for MVS & VM is designed to provide defined calling conventions, enhanced interlanguage communication, callable services, language-specific services (for example, common facilities messages), common math functions, application utilities, system services, and subsystem support for Customer Information Control System (CICS) and Information Management System (IMS)

With Language Environment for MVS & VM, application programmers can extend and integrate their applications and packages, as well as reuse code with greater flexibility, due to interlanguage communication and native language restrictions. Application packages may be extended with the language of choice. Language Environment for MVS & VM enables existing applications to function as before with little, if any, changes required, thus helping preserve company investment in those applications. Language Environment for MVS & VM condition handler permits programs to handle errors in a predictable, logical manner without highly-specialized routines

Language Environment for MVS & VM replaces the existing language-specific run-time libraries and provides a common runtime environment for all languages that conform to the Language Environment architecture. In a single product, Language Environment for MVS & VM combines essential run-time services, such as routines for message handling, condition handling, and storage management. All these services are available through a set of interfaces that are consistent across programming languages. With Language Environment for MVS & VM, application programmers can use one runtime environment for their applications, regardless of the programming language or system resource needs

Today, IBM Language Environment for MVS & VM is the run-time library for the Language Environment-enabled compilers IBM COBOL for MVS & VM, C/C++

for MVS/ESA, and PL/I MVS & VM. IBM realizes the importance our customers place on complete, open, high-quality solutions. Therefore, IBM has established an *IBM Language Environment Partner Program* at the IBM Santa Teresa Laboratory. This program encourages and assists tool and application package vendors to support and take advantage of Language Environment services. Companies with existing 3GL applications will benefit from the wider choice of tools and application packages.

Language Environment for MVS & VM provides a number of advantages over other packages, its capabilities include:

- Ability to parse dates in an infinite number of formats by using a picture string feature as a parsing guide
- Provide NLS support by using a built-in table of defaults based on a country code
- A sliding window feature

  IBM Language Environment for MVS & VM provides a valuable short term solution with the century window feature. If you are unable to change all of your applications and data at the same time, the century window feature allows 2-digit years to be interpreted in a 100-year window. Any 100-year window can be selected. You pass a 2-digit year to Language Environment and Language Environment returns a 4-digit year based on the 100-year window.

  The advantage to the century window is that you need to change only the application code and not the databases or files with 2-digit years. This allows you to change the application programs one at a time or groups at a time without effecting your data files. Note, this only works for dates that range less than 100 years. For example, dates of birth may not be appropriate for this solution; a person born in '94' could be over 100 years of age.

  The disadvantage to the century window is that it doesn't last forever. If your application has a long life expectancy, you many need to go back and replace the century window with full 4-digit year support. Some of your applications will be replaced prior to this replacement.

IBM CoOperative Development Environment/370 (CODE/370)

IBM CoOperative Development Environment/370 (CODE/370) (program product # 5688-194) — the common editor, compiler, debug tool — provides a cooperative environment, allowing application programmers to more productively develop and maintain host IBM 3GL applications from the workstation. CODE/370 provides a consistent, graphical user interface across different platforms and languages, a language-sensitive editor, language-sensitive help, a compiler invocation facility, and an interactive debug tool. CODE/370 combines the richness of the S/370 or S/390 subsystem environments and the power of IBM Language Environment for MVS & VM to provide a host Debug Tool which allows programmers to find bugs, fix bugs, and test applications. The Debug Tool is available either as a stand-alone 3270 host debug tool (for programming shops where workstations are not available) or with an optional graphical workstation user interface.

CODE/370's cooperative environment allows application programmers to perform host programming tasks, such as compiling and debugging, from a workstation. Through cooperative processing, users perform functions locally on the workstation while interactively accessing the programs, data, and compilers residing on a host system The optional workstation interface combines the Edit and Compile / Link functions together with the Debug Tool graphical user interface (GUI)

The powerful workstation-based editor integrates a rich set of functions that will speed up your application-development activities The editor works with any type of source The language-sensitive features help optimize coding efficiency in COBOL, C, PL/I, REXX, and JCL Source can be stored in an MVS data set, a VM file, or an OS/2 file As you edit source in any of these formats, CODE/370 maintains the sequence numbers and date stamps.

This method of workstation/host tool integration offers the best use of the two environments. the S/370 subsystem environments' "live" debug capabilities and the workstation GUI's easy editing capabilities The integration also allows programming shops to grow at their own pace into developing host applications on workstations

When programmers use the Debug Tool, the debug session is recorded in a log file, permitting edit and replay of a Debug Tool session This allows the Debug Tool to be used to capture test cases (for future program validation) or to further isolate a problem within an application This also allows both interactive and batch debugging of a programmer's application.

CODE/370 provides

- Support for COBOL/370 Rel 1, COBOL for MVS & VM Rel 2, C/370 and PL/I MVS & VM
- Limited support for VS COBOL II Rel 3 1, 3 2, and 4 0 and OS PL/I Ver 2 Rel 1, 2, and 3
- A 32-bit, user-programmable editor
- Several language-sensitive editing features for REXX and JCL
- Enhanced support for debugging under CICS
  - Pseudo conversational transaction support for COBOL applications
  - Support for the full range of single-terminal Send and Receive messages
- Program Generator, an independent compile/link program that allows compiling from inside and outside the editor
- REXX and JCL programs can be submitted to the host from the Editor window
- An OS/2 desktop tool called WorkFrame/2
- Advanced Program to Program Communication (APPC) protocol support for cooperative sessions between the MVS host and the workstation
- Support for debugging COBOL applications consisting of multiple enclaves and multiple processes
- Debug Tool support for exception handling of COBOL IGZ exceptions
- The ability to perform initial installation of CODE/370's workstation feature from a LAN server

Other Host PL/I Compilers for MVS & VM

IBM has developed three PL/I products for the MVS and VM mainframe OS PL/I V1, OS PL/I V2, and PL/I for MVS & VM

OS PL/1 V1 will be withdrawn from service in December of 1995 This section discusses issues involved with migrating from OS PL/I V1 The two PL/I mainframe products that will continue to be available are PL/I for MVS & VM and OS PL/I V2

Because OS PL/I V1 (5734-PL1, 5734-PL2, 5734-PL3, 5734-LM4) will be discontinued, we are encouraging companies to upgrade their PL/I technology to PL/I for MVS & VM (compiler) with Language Environment for MVS & VM (run-time library) Depending on which product is currently being used and how fast a company is willing to migrate, there are three possible migration paths that can be followed

Migration Paths: The three paths are

1. OS PL/I V1------------------→PL/I for MVS & VM

   Companies with OS PL/I V1 are encouraged to migrate directly to IBM PL/I for MVS & VM (compiler) and IBM Language Environment for MVS & VM (run-time library) if the IBM Language Environment for MVS & VM prerequisites are satisfied See the IBM Language Environment for MVS & VM Licensed Program Specification for prerequisite information

2. OS PL/I V1----→OS PL/I V2----→PL/I for MVS & VM

   Companies with OS PL/I V1 who do not yet satisfy the IBM Language Environment for MVS & VM prerequisites must migrate to OS PL/I V2 first. When the IBM Language Environment for MVS & VM prerequisites are satisfied, then move to IBM PL/I for MVS & VM

3. OS PL/I V2----→PL/I for MVS & VM

   Companies with OS PL/I V2 can migrate to IBM PL/I for MVS & VM once the IBM Language Environment for MVS & VM prerequisites are satisfied.

EDGE Portfolio Analyzer - PL/I Migration Tool

The EDGE Portfolio Analyzer Version 1 (program product # 5633-009) can significantly reduce the effort necessary to migrate from an earlier version of an IBM host PL/I compiler to the latest level of this product.

This migration tool analyzes existing application load libraries and provides data and statistical information about

- Which release of a compiler and what compiler and linkage editor options are used to produce an existing load module
- Application programs
  - Containing shared subroutines
  - Requiring re-linking of runtime modules as a result of any arbitrary change
  - Impacted by implementation of a new runtime package
  - Not expected to be affected by the implementation of a new runtime package
  - Containing interlanguage dependencies that may prove obstacles to migration
- Load modules not conforming to the installation's established compiler and linkage editor standards

Customers can save many hours of tedious investigation and move more quickly into the new compiler products

Workstation Interactive Test Tool (WITT)

Workstation Interactive Test Tool (WITT) Product Family currently consists of WITT for OS/2 (program product # 5648-031), WITT for AIX (program product # 5765-222), WITT for Windows (program part # 10H7744), WITT for SUN (program product # 5765-407), and WITT for HP (program product # 5765-406).

COBOL programmers can use the Workstation Interactive Test Tool (WITT) as a GUI test tool that automatically facilitates the creation of reusable unit test, function test, system test, and regression test cases. These reusable test cases can be created when the program is newly developed and should be run continuously during the years that the program is maintained. Once created, the test cases can be played back interactively or in an unattended batch mode The execution of a set of WITT test cases provides an audit trail of errors that a tester can provide to a programmer to document program failures

WITT automatically records and plays back all keystroke and mouse movements and compares and identifies test inconsistencies between benchmark test cases and test cases run after enhancements and fixes are made to a program WITT also allows scripting in 2/REXX to add program intelligence to the test cases This makes WITT test cases easy to update, maintain, and reuse for future testing

WITT/OS2 installs on an OS/2 desktop and allows the testing of MVS, VM, IMS, CICS, OS/400, and OS/2 PM and Text (ie, Micro Focus, CICS/OS2) applications WITT/PM installs on an OS/2 desktop and allows the testing of only OS/2 PM and Text applications. X/WITT installs on an AIX desktop and allows the testing of AIX X Window applications, as well as remote client applications in X Windows environments on SUN, HP, APOLLO, and MVS WITT/Windows installs on a Windows 3.1 desktop and allows the testing of Windows workstation applications

SORT

DFSORT V1R13 will enhance its Year2000 capabilities by providing the ability to sort, merge, and transform 2-digit years according to a specified sliding or fixed century window New Y2C, Y2Z, Y2P, and Y2D formats, in conjunction with a new Y2PAST installation and run-time option, allow you to handle 2-digit year data in the following ways

- Set the appropriate century window for your applications For example, set a century window of 1915-2014 or 1950-2049
- Order 2-digit character, zoned decimal, packed decimal, or decimal year data, according to the century window, using DFSORT's SORT and MERGE control statements For example, order 96 (representing 1996) before 00 (representing 2000) in ascending sequence, or order 00 before 96 in descending sequence.
- Transform 2-digit character, zoned decimal, packed decimal, or decimal year data to 4-digit character year data, according to the century window, using DFSORT's OUTFIL control statement. For example, transform 99 to 1999 and 04 to 2004

These DFSORT enhancements allow you to continue to use 2-digit years for sorting and merging, and assist those situations when you want to change 2-digit-year data to 4-digit-year data.

Additional information about DFSORT/MVS and its year2000 enhancements is available on the World Wide Web at URL

`http://www.storage.ibm.com/storage/software/sort/srtmhome.htm`

Sliding Century Window

A new installation and run-time option allows you to specify a sliding or fixed century window to be used with 2-digit years. `Y2PAST=s` specifies a **sliding** century window starting s years before the current year. For example, if the current year is 1996, Y2PAST=80 starts the century window at 1996 – 80 = 1916, providing a century window of 1916 through 2015. In 1997, this century window automatically slides to 1917 through 2016.

`Y2KPAST=f` specifies a **fixed** century window starting at f. For example, Y2KPAST=1950 starts the century window at 1950, providing a century window of 1950 through 2049. Thus, Y2PAST allows you to control how DFSORT interprets the 2-digit years 00-99 on a site-wide or application-specific basis.

As an example, both Y2KPAST=1915 and Y2PAST=81 used in 1996 give a century window of 1915 through 2014, and result in the following interpretation of 2-digit year formatted data by DFSORT:

yy	**Interpreted as:**
00	2000
14	2014
15	1915
61	1961
62	1962
99	1999

2-Digit Year Formats

New formats allow you to identify 2-digit character, zoned decimal, packed decimal and decimal year data for special DFSORT processing as follows. (yy represents 2-digit year data in the examples below.)

Format	Meaning
Y2C	identifies 2-digit, 2-byte character year data such as C'yy', C'mm/dd/yy', or C'yy.mm.dd'.
Y2Z	identifies 2-digit, 2-byte zoned decimal year data such as Z'yy', Z'mmddyy', or Z'yymmdd'.
Y2P	identifies 2-digit, 2-byte packed decimal year data such as P'yy', P'dddyy', or P'yymmdd'.
Y2D	identifies 2-digit, 1-byte decimal year data such as X'yy' or P'yyddd'.

Sorting and Merging 2-Digit Years

You can use the new Y2C, Y2Z, Y2P, and Y2D formats in DFSORT's SORT and MERGE statements to identify specific 2-digit year data to be ordered according to the century window.

A simple example of the control statements to sort a C'mm/dd/yy' field (assume the current year is 1996) follows:

```
* Set the century window to 1962 through 2061
  OPTION Y2PAST=34
* Sort C'mm/dd/yy' as C'yymmdd'
  SORT FIELDS=(7,2,Y2C,A, * sort yy using century window
             1,2,CH,A,    * sort mm
             4,2,CH,A)    * sort dd
```

These control statements provide the following sort results

Input Data (CH)	Sorted Output Data (CH)
06/22/15	03/18/62
10/03/00	09/01/99
11/14/61	10/03/00
08/16/14	08/16/14
09/01/99	08/17/14
03/18/62	06/22/15
08/17/14	11/14/61

Transforming 2-Digit Years to 4-Digit Years

You can use the new Y2C, Y2Z Y2P, and Y2D formats in the OUTREC operand of DFSORT's OUTFIL statement to identify 2-digit year data to be changed to 4-digit year data according to the century window

A simple example of the control statements to transform a P'yyddd' field follows

```
* Set the century window to 1970 through 2069
  OPTION COPY,Y2PAST=1970
* Change P'yyddd' to C'yyyy/ddd'
 OUTFIL FNAMES=Y4,
  OUTREC=(1,1,Y2D,      * change X'yy' to C'yyyy' using
*                       *   century window
          C'/',         * insert C'/'
          2,2,PD,M11)   * change P'ddd' to C'ddd'
```

This code provides the following transformation results

Input Data (HEX)	Transformed Output Data (CH)
92012F	1992/012
70225C	1970/225
69153F	2069/153
00001F	2000/001
99321F	1999/321
12054C	2012/054

COMUDAS (COMmon Uithoorn DAte Services)

COMUDAS (program product # 5788-HBB) is a common date routine that has been developed to replace all existing date routines, used by the IBM Uithoorn Lab, The Netherlands COMUDAS offers all necessary functions for validation, conversion, and calculation of dates in any format A standard interface must be used to call the date routine's load module dynamically

The package also offers the possibility to use separate functions by means of NCAL's, that can be linked statically This might be useful when converting large files or databases with validated data into other formats (for example, when reformatting year-date notation in an application)

Functions are available for updating the Calendar Tables by means of the Online Facility, that also can be used to test the Date Routines, and to print calendars

A CICS version, as well as an MVS version, of this package is available

Features

COMUDAS provides features including

- Date validation
- Date calculation
- Date conversion
- Free date-format definitions
- Supports country-dependent data
  - Weekend definitions
  - Closing dates (for both manufacturing and fiscal purposes)
  - Holidays
- Can be used by MVS applications with PL/1 runtime environment or a Language Environment, for
  - CICS (PL/1)
  - PL/1
  - COBOL
- Supports terms as
  - Country codes
  - Shopdates (numbering of working days)
  - Production months
  - Closing dates (logistics/manufacturing & finance)
- Is able to print
  - Common calendars for years in range 1760 through 9999
  - Production calendars (per country/year)
- Contains functions to support a calendar owner
- Can be used as a tool when changing applications to handle the year 2000

Components

COMUDAS consists of two main components First is the date routine functions An interface is supplied that can be filled with data, and returned to the calling program, containing the requested information, including a return code and return message The interface is available in both PL/1 and COBOL format The date routine functions can be called by applications

- CICS (PL/1), for example, HPS applications
- PL/1
- COBOL

The second component, the online facility, used in a TSO/ISPF environment, can be used to:

- Get familiar with the date routines

  An online presentation of the interface can be used to test the date routine functions
- Update the calendar tables

  All users (with read access) may view the contents of the calendar tables, and the calendar owner can use this function to
  - Update the calendar tables
  - Generate a new table to be used by the date routine functions
  - Generate production calendars
- Print a calendar

  All users will be able to print a common calendar for years in range 1760 through 9999
- Print a Production Calendar

  All users will be able to print a production calendar for a country and year, that must have been created by the calendar owner previously

Using COMUDAS

When using COMUDAS, the following terms can be handled by the package

- Day of the week (number, name, 3-character abbreviation)
- Day of the month (number)
- Day of the year (Julian notation)
- Week of the year (number)
- Production Month (number)
- Month of the year (number, name, 3-character abbreviation)
- Year (two or four digits)
- Year (related to a weeknumber, this value may differ from the earlier mentioned year-field, because weeks do not always start on January 1)
- Shopdates
- SYSDATE
- European format of a date
- ISO (International Standards Organization) format of a date
- USA format of a date
- JIS (Japanese Industrial Standard) format of a date
- Remainder days of a subtraction of two dates
- Day number suffix (for example, 10th, 23rd)
- First shopdate of a week
- Last shopdate of a week
- Startdate of a production month
- Closing date of a production month
- Closing date finance
- Addition / Subtraction fields
  - +/- days
  - +/- weeks
  - +/- weeks and days
  - +/- months
  - +/- months and days
  - +/- years, months and days
  - +/- shopdays

Special Ordering Information

To order COMUDAS, your IBM marketing representative must order the package through a formal note to

Vnet COMUDAS at UITVM1

IBM Tools for VM

The IBM COBOL Family for MVS & VM

For application code written in COBOL for S/370 and S/390 (for MVS and VM) platforms, IBM has developed COBOL compilers These products target the host application development environment

IBM's COBOL mainframe products for S/370 and S/390 are

- **COBOL for MVS & VM** (new name for IBM COBOL/370) — the compiler
- **Language Environment for MVS & VM** (new name for Language Environment/370) — the run-time library
- **CoOperative Development Environment** (CODE/370) — edit/compile/debug tool

See "The IBM COBOL Family for MVS & VM" on page 7-7 for detailed information about these COBOL tools

The IBM PL/I Family for MVS & VM

For application code written in PL/I for S/370 and S/390 (for MVS and VM) platforms, IBM has developed PL/I compilers These products target the host application development environment.

IBM's PL/I mainframe products for S/370 and S/390 are

- **PL/I for MVS & VM** — the compiler
- **Language Environment for MVS & VM** (new name for Language Environment/370) — the run-time library
- **CoOperative Development Environment** (CODE/370) — edit/compile/debug tool

See "The IBM PL/I Family for MVS & VM" on page 7-18 for detailed information about these PL/I tools

REXX/EXEC Migration Tool for VM/ESA

The REXX/EXEC Migration Tool for VM/ESA (VM/ESA MIGR) is a tool that helps migrate REXX, EXEC2, ASSEMBLE, and other source files to the current release of VM/ESA

VM/ESA MIGR does not make changes, but can assist the user in the following areas

- Estimating the migration effort that is necessary
- Identifying changes that have to be made
- Finding commands, options, etc which are incompatible or have been changed and presenting information about them through Help panels

VM/ESA MIGR uses a keyword file (ESAMIGR SAMPLIST) to flag items which may cause incompatibilities ESAMIGR SAMPLIST may be customized by the user to search for additional items or variations (such as command abbreviations) of items

VM/ESA MIGR is shipped on a separate tape with the VM/ESA product. It is also available on MKTTOOLS, and through PUBORDER. Complete documentation is available in *REXX/EXEC Migration Tool for VM/ESA*, (GC24-5607)

IBM Tools for VSE

The IBM COBOL Family for VSE

For application code written in COBOL for S/370 and S/390 for VSE platform, IBM has developed COBOL compilers. These products target the host application development environment

IBM's COBOL mainframe products for S/370 and S/390 VSE are

- **COBOL for VSE/ESA** — the compiler
- **Language Environment for VSE/ESA** — the run-time library

Features

This COBOL compiler, COBOL for VSE/ESA, is available today and provides full 4-digit year support with features including:

- Intrinsic functions
- Sliding century window

COBOL for VSE/ESA provides full Year2000 support. It provides ANSI COBOL Standard Intrinsic Functions which give full date manipulation capability with 4-digit-year support Language Environment for VSE/ESA provides additional date manipulation with the Language Environment Callable Services COBOL for VSE/ESA will return a 4-digit year when the COBOL application program queries the system for the current date. DOS/VS COBOL and VS COBOL II return the system current date with a 2-digit year.

Companies need productive application development environments A language compiler is only a portion of what application programmers need today to develop and maintain code Tools that generate statistics from code, modularize, migrate, or restructure code, or search class libraries, are becoming more and more important

These tools assist in migrating your existing COBOL applications to COBOL for VSE/ESA

- **CCCA for VSE** — converts source code
- **COBOL Report Writer Precompiler** — supports Report Writer code

IBM COBOL for VSE/ESA

IBM COBOL for VSE/ESA (program product # 5686-068) is the high-performance compiler with a supporting runtime environment (Language Environment for VSE/ESA (program product # 5686-067), which facilitates multiple-language interaction. This is different from the single packaging of compiler and runtime environment for DOS/VS COBOL and VS COBOL II

Applications written using COBOL for VSE/ESA can interface with a variety of IBM products, such as SQL/DS and CICS The consistent interlanguage communication support, common protocols, and suite of callable services provided by Language Environment for VSE/ESA is designed to allow easier access to in-house applications or vendor packages written in COBOL for VSE/ESA

IBM COBOL for VSE/ESA provides:

- Intrinsic functions, which reduce the need for extensive algorithms

- Programmer access to all of the elements in a table at once, reducing the need for explicit loops
- Consistent interlanguage communications, common services, and common functions, which helps extend the useful life of existing applications
- Improved dynamic calls
- Support for Year2000
- Capabilities to help application programmers incrementally enhance applications
- Help in maintaining and enhancing the investment in existing programmer skills

IBM COBOL for VSE/ESA provides facilities to acquire and integrate packaged software, consistent with the vendor's terms, into existing applications irrespective of the language used, use existing code in new applications (code reuse) regardless of the source code language used, and invoke functionality between applications with improved interlanguage communication (ILC)

IBM COBOL for VSE/ESA includes VS COBOL II language features, enhanced compiler options, virtual storage constraint relief, structured programming language for improved programmer productivity enhanced double-byte character set (DBCS) support, streamlined system interfaces, expanded code optimization, flexible run-time options, and support for the VSE/ESA 31-bit addressing feature

Most applications were coded with 2-digit-year data COBOL for VSE/ESA provides intrinsic functions not available in our earlier COBOL compilers Calendar functions include

- CURRENT-DATE
- DATE-OF-INTEGER
- DAY-OF-INTEGER
- INTEGER-OF-DATE
- INTEGER-OF-DAY
- WHEN-COMPILED

Example of obtaining the 4-digit year with the COBOL Intrinsic Functions

- DATE-OF-INTEGER gives YYYYMMDD
- DAY-OF-INTEGER gives YYYYDDD

IBM Language Environment for VSE/ESA

IBM Language Environment for VSE/ESA (program product # 5686-067) is IBM's common runtime environment for enterprise applications written in COBOL and PL/I. Language Environment for VSE/ESA is designed to provide defined calling conventions, enhanced interlanguage communication, callable services, language-specific services (for example, common facilities messages), common math functions, application utilities, system services, and subsystem support for Customer Information Control System (CICS).

With Language Environment for VSE/ESA, application programmers can extend and integrate their applications and packages, as well as reuse code with greater flexibility, due to interlanguage communication and native language restrictions Application packages may be extended with the language of choice Language Environment for VSE/ESA enables existing applications to function as before with little, if any, changes required, thus helping preserve company investment in those applications Language Environment for VSE/ESA condition handler permits programs to handle errors in a predictable, logical manner without highly-specialized routines

Language Environment for VSE/ESA replaces the existing language-specific run-time libraries and provides a common runtime environment for all languages that conform to the Language Environment architecture. In a single product, Language Environment for VSE/ESA combines essential run-time services, such as routines for message handling, condition handling, and storage management All these services are available through a set of interfaces that are consistent across programming languages. With Language Environment for VSE/ESA, application programmers can use one runtime environment for their applications, regardless of the programming language or system resource needs

Today, IBM Language Environment for VSE/ESA is the run-time library for the Language Environment-enabled compilers IBM COBOL for VSE/ESA and PL/I for VSE/ESA IBM realizes the importance our customers place on complete, open, high-quality solutions Therefore, IBM has established an *IBM Language Environment Partner Program* at the IBM Santa Teresa Laboratory This program encourages and assists tool and application package vendors to support and take advantage of Language Environment services Companies with existing 3GL applications will benefit from the wider choice of tools and application packages.

Language Environment for VSE/ESA provides a number of advantages over other packages, its capabilities include:

- Ability to parse dates in an infinite number of formats by using a picture string feature as a parsing guide
- Provide NLS support by using a built-in table of defaults based on a country code
- A sliding window feature

  IBM Language Environment for VSE/ESA provides a valuable short term solution with the century window feature If you are unable to change all of your applications and data at the same time, the century window feature allows 2-digit years to be interpreted in a 100-year window Any 100-year window can be selected. You pass a 2-digit year to Language Environment and Language Environment returns a 4-digit year based on the 100-year window

  The advantage to the century window is that you need to change only the application code and not the databases or files with 2-digit years This allows you to change the application programs one at a time or groups at a time without effecting your data files Note, this only works for dates that range less than 100 years For example, dates of birth may not be appropriate for this solution, a person born in '94' could be over 100 years of age.

  The disadvantage to the century window is that it doesn't last forever If your application has a long life expectancy, you many need to go back and replace the century window with full 4-digit year support Some of your applications will be replaced prior to this replacement

Other Host COBOL Compilers for VSE

IBM has developed three COBOL products for the VSE environment DOS/VS COBOL, VS COBOL II, and COBOL for VSE/ESA This section discusses issues involved with migrating from DOS/VS COBOL.

Migrating from DOS/VS COBOL: We are encouraging companies to upgrade their COBOL technology to COBOL for VSE/ESA (compiler) with Language Environment for VSE (run-time library) There are three possible migration paths that you can follow

Migration Paths: The paths are

1 DOS/VS COBOL----------------→COBOL for VSE/ESA

Companies with DOS/VS COBOL are encouraged to migrate directly to IBM COBOL for VSE/ESA (compiler) and IBM Language Environment for VSE (run-time library) if the prerequisites are satisfied

2 DOS/VS COBOL---→VS COBOL II---→COBOL for VSE/ESA

Companies with DOS/VS COBOL who do not yet satisfy the IBM Language Environment for VSE/ESA prerequisites should migrate to VS COBOL II first When the IBM Language Environment for VSE/ESA prerequisites are satisfied, then a company can move to IBM COBOL for VSE/ESA

Each successive COBOL product contains more capabilities than the previous product

VS COBOL II: VS COBOL II (program product # 5668-958) builds on the functions of DOS/VS COBOL but has a variety of features that give companies many advantages over earlier IBM COBOL products. VS COBOL II includes additional language features, enhanced compiler options, virtual storage constraint relief, structured programming language for improved programmer productivity, enhanced double-byte character set (DBCS) support, streamlined system interfaces, expanded code optimization, flexible run-time options, and support for VSE/ESA's 31-bit addressing feature

Migration Path:

VS COBOL II---→COBOL for VSE/ESA

Companies with VS COBOL II can migrate to IBM COBOL for VSE/ESA once the IBM Language Environment for VSE/ESA prerequisites are satisfied

Customers who have VSE/ESA Version 2 1 or 1 4 installed and meet the prerequisites should migrate to IBM COBOL for VSE/ESA and Language Environment for VSE/ESA

Benefits of COBOL Migration: DOS/VS COBOL is the ANSI 74 compiler ANSI 85 introduced many significant functions which are provided to customers in either VS COBOL II or COBOL for VSE/ESA Customers who have migrated to the newer COBOL Standard have additional functionality, increased developer productivity and exploitation of S/390 hardware capabilities Some benefits of upgrading COBOL technology are

- Improved Interlanguage Communication (ILC)
- Condition management features of Language Environment, which bring PL/I-like condition handling to COBOL

- Language Environment callable services, including a date/time service routine that interprets a 2-digit year to a 4-digit year to accommodate the year 2000
- Improved application performance of COBOL for VSE/ESA compared to VS COBOL II

For more information on the value of migrating from DOS/VS COBOL or VS COBOL II to COBOL for VSE/ESA, obtain a copy of ***Why Migrate to COBOL/370 and LE/370?***, which is available from your IBM representative (COBMGVAL PACKAGE on MKTTOOLS). This document contains some examples of customer benefits of migrating to COBOL/370 that also applies to the VSE environment

COBOL and CICS/VS Command Level Conversion Aid (CCCA) for VSE

COBOL and CICS/VS Command Level Conversion Aid (CCCA) for VSE (program product # 5785-CCC) is an effective tool designed to make it easier to convert old COBOL source code and copy modules to the new COBOL Standard CCCA converts DOS/VS COBOL and COBOL 74 Standard VS COBOL II (Release 3 and 4 (CMPR2)) source code to COBOL 85 Standard VS COBOL II Release 3 or 4 (NOCMPR2) or to IBM COBOL for VSE/ESA

In cases where a statement is no longer supported and has no equivalent statement in the target COBOL, CCCA flags the statement CCCA can be used to convert from DOS/VS COBOL to COBOL for VSE/ESA, just as it is used to convert from DOS/VS COBOL to VS COBOL II. The source file output for compiling under VS COBOL II can also be used for compiling under COBOL for VSE/ESA

CCCA is designed to identify and convert source code incompatibility, to reduce the effort required to convert programs, and to minimize conversion errors The conversion process can be customized by users to meet unique conversion requirements. Installation and usage are easy, fast, and straightforward

CCCA key benefits are

- Identification and conversion of source code
- Reduction of the effort required to convert programs
- Minimization of conversion errors
- Enhanced programmer productivity during migration

CCCA provides facilities to

- Convert most syntax differences between OS/VS COBOL, DOS/VS COBOL, or VS COBOL II Release 1 or 2 and the current release of VS COBOL II and COBOL for MVS & VM and COBOL for VSE/ESA programs
- Convert EXEC CICS commands
- Remove and/or convert the base locator for linkage (BLL) section mechanism and references
- Eliminate conflicts between user-defined names and words reserved for VS COBOL II
- Convert both source programs and copy modules
- Create conversion management reports
- Produce a statement-by-statement diagnostic listing showing the result of the conversion process for each program
- Change and/or create COBOL conversion modules
- Allow foreground conversion of CICS programs

- Perform conversion from various levels of COBOL into other COBOL levels through an open converter design
- Read from PDSEs, not just PDSs

COBOL Report Writer Precompiler

The COBOL Report Writer Precompiler (program offering # 5798-DYR) has two functions It can permanently convert Report Writer statements to valid COBOL statements that can be compiled in IBM COBOL for MVS & VM or IBM COBOL for VSE/ESA Or, it can be used to precompile applications containing Report Writer statements so the code will be acceptable to the IBM COBOL for MVS & VM or IBM COBOL for VSE/ESA compiler

When used to precompile, the Precompiler automatically invokes the IBM COBOL compiler—as though Report Writer statements in the source program are being processed by the IBM COBOL for MVS & VM or IBM COBOL for VSE/ESA compiler itself The fact that two separate processes are involved is transparent to users

The IBM PL/I Family for VSE

For application code written in PL/I for S/370 and S/390 for VSE platform, IBM has developed PL/I compilers These products target the host application development environment

IBM's PL/I mainframe products for S/370 and S/390 VSE are

- **PL/I for VSE/ESA** — the compiler
- **Language Environment for VSE/ESA** — the run-time library

Features

This PL/I compiler, PL/I for VSE/ESA, is available today and provides full 4-digit year support with features including

- Built-in functions
- Sliding century window

PL/I for VSE/ESA provides full Year2000 support It provides Built-in Function which provides 4-digit-year support Language Environment for VSE/ESA provides additional date manipulation with the Language Environment Callable Services PL/I for VSE/ESA will return a 4-digit year when the PL/I application program queries the system for the current date

IBM PL/I for VSE/ESA

IBM PL/I for VSE (program product # 5686-069) is an implementation of IBM PL/I for MVS & VM and succeeds DOS PL/I for VSE - providing source code compatibility for most DOS PL/I for VSE R6 programs PL/I for VSE requires and takes advantage of Language Environment for VSE capabilities to support use in mixed-language development environments and the use of reusable components in building applications IBM PL/I for VSE provides

- 31-bit virtual addressing
- Support for Language Environment for VSE functions
- Enhanced InterLanguage Communication (ILC) with COBOL for VSE

IBM Language Environment for VSE/ESA

IBM Language Environment for VSE/ESA (program product # 5686-067) is IBM's common runtime environment for enterprise applications written in COBOL and PL/I. Language Environment for VSE/ESA is designed to provide defined calling conventions, enhanced interlanguage communication, callable services, language-specific services (for example, common facilities messages), common math functions, application utilities, system services, and subsystem support for Customer Information Control System (CICS).

With Language Environment for VSE/ESA, application programmers can extend and integrate their applications and packages, as well as reuse code with greater flexibility, due to interlanguage communication and native language restrictions. Application packages may be extended with the language of choice. Language Environment for VSE/ESA enables existing applications to function as before with little, if any, changes required, thus helping preserve company investment in those applications. Language Environment for VSE/ESA condition handler permits programs to handle errors in a predictable, logical manner without highly-specialized routines.

Language Environment for VSE/ESA replaces the existing language-specific run-time libraries and provides a common runtime environment for all languages that conform to the Language Environment architecture. In a single product, Language Environment for VSE/ESA combines essential run-time services, such as routines for message handling, condition handling, and storage management. All these services are available through a set of interfaces that are consistent across programming languages. With Language Environment for VSE/ESA, application programmers can use one runtime environment for their applications, regardless of the programming language or system resource needs.

Today, IBM Language Environment for VSE/ESA is the run-time library for the Language Environment-enabled compilers IBM COBOL for VSE/ESA and PL/I for VSE/ESA. IBM realizes the importance our customers place on complete, open high-quality solutions. Therefore, IBM has established an *IBM Language Environment Partner Program* at the IBM Santa Teresa Laboratory. This program encourages and assists tool and application package vendors to support and take advantage of Language Environment services. Companies with existing 3GL applications will benefit from the wider choice of tools and application packages.

Language Environment for VSE/ESA provides a number of advantages over other packages; its capabilities include:

- Ability to parse dates in an infinite number of formats by using a picture string feature as a parsing guide
- Provide NLS support by using a built-in table of defaults based on a country code
- A sliding window feature

  IBM Language Environment for VSE/ESA provides a valuable short term solution with the century window feature. If you are unable to change all of your applications and data at the same time, the century window feature allows 2-digit years to be interpreted in a 100-year window. Any 100-year window can be selected. You pass a 2-digit year to Language Environment and Language Environment returns a 4-digit year based on the 100-year window

The advantage to the century window is that you need to change only the application code and not the databases or files with 2-digit years This allows you to change the application programs one at a time or groups at a time without effecting your data files Note, this only works for dates that range less than 100 years For example, dates of birth may not be appropriate for this solution, a person born in '94' could be over 100 years of age

The disadvantage to the century window is that it doesn't last forever If your application has a long life expectancy, you many need to go back and replace the century window with full 4-digit year support. Some of your applications will be replaced prior to this replacement

IBM Tools for AS/400

You can use the IBM products and tools listed below to help you address the changes required as you proceed with your Year2000 transition The tools listed, however, represent only a subset of the AS/400 oriented offerings available to you for application development. A more complete list of available tools can be found in the *AS/400 Application Development Handbook*, (G325-6249)

The IBM RPG Family

For application code written in RPG, IBM has developed several RPG language compilers These compilers target the host application development environment

IBM's RPG compilers for AS/400 are

- Integrated Language Environment (ILE) RPG IV
- Original Program Model (OPM) RPG/400
- System/36 compatible compiler

All three RPG language compilers are available with either the Integrated Language Environment RPG for OS/400 Version 3 product (5716-RG1) or the IBM Integrated Language Environment RPG/400 Version 3 product (5763-RG1)

Using ILE RPG IV Date Support: The ILE RPG IV compiler has the following support for dates.

- Retrieval of the job date through special words UDATE and UYEAR This level of support provides for a 2-digit year
- Retrieval of the job date through special words *DATE and *YEAR This level of support provides for a 4-digit year
- Retrieval of the system date through operation TIME This level of support provides for both 2-digit and 4-digit years
- Date, Time, and Timestamp data type support This level of support provides for date operations such as subtraction, addition, extraction, testing, comparision, and move
- Calling OPM System APIs This level of support provides for both 2-digit and 4-digit years
- Calling ILE Date APIs This level of support provides for both 2-digit and 4-digit years
- SQL date, time, and timestamp data type support This level of support provides for 4-digit years

This section provides information on how to use RPG date, time, and timestamp fields Specifically, it covers

- The date, time, and timestamp data types and their formats
- User date special words
- How to edit date-time fields
- Date-time keywords
- Date-time operations and how to use them

Date-Time Data Types and Formats

Date, time and timestamp fields have an internal format that is independent of the external format The **internal format** is the way the data is stored in the program The **external format** is the way the data is stored in files

You need to be aware of the internal format when:

- Passing parameters
- Overlaying subfields in data structures

There is a default internal format for date, time, and timestamp fields. In general, to change the internal format for a specific field, you must define the field and specify its internal format on a Definition specification Similarly, to change (or specify) the external format for program-described fields, you specify the format on the corresponding Input or Output specificiation

For fields in an externally described file, the external data format is specified in the data description specifications in position 35 You cannot change the external format of externally described date, time, and timestamp fields

For subfields in externally described data structures, the data formats specified in the external description are used as the internal formats of the subfields by the compiler The reason for this difference, is that a data structure, even if externally described, exists only when a program is running

Internal Format: The default format for date, time, and timestamp fields is *ISO In general, it is recommended that you use the default ISO internal format, especially if you have a mixture of external format types.

For date, time, and timestamp fields, you can change the default format in two ways You can use the DATFMT and TIMFMT keywords on the Control specification to change the default internal format, if desired, for *all* date-time fields in the program In addition, you can use the Definition specification to

- Override the default internal format by using the DATFMT and TIMFMT keywords
- Specify an initial value for a date, time, or timestamp field that is different than the default, by using the INZ keyword

Specifying an External Format for a Date-Time Field: If you have date, time, and timestamp fields in program-described files, then you *must* specify their external format You can specify a default external format for all date, time, and timestamp fields in a program-described file by using the DATFMT and TIMFMT keywords on a File-Description specification You can specify an external format for a particular field as well. Specify the desired format in positions 31-34 on an Input specification. Specify the appropriate keyword and format in positions 53-80 on an Output specification

For more information on each format type, see the appropriate section in the remainder of this chapter

Date Data Type: Date fields have a predetermined size and format They can be defined on the definition specification Leading and trailing zeros are required for all date data

Date constants or variables used in comparisons or assignments do not have to be in the same format or use the same separators Also, dates used for I/O operations such as input fields, output fields or key fields are also converted (if required) to the necessary format for the operation.

The default internal format for date variables is *ISO. This default internal format can be overridden globally by the control specification keyword DATFMT and individually by the definition specification keyword DATFMT.

The hierarchy used when determining the internal date format and separator for a date field is:

1. From the DATFMT keyword specified on the definition specification
2. From the DATFMT keyword specified on the control specification
3. *ISO

There are two kinds of date data formats: 2-digit and 4-digit-year formats. For 2-digit-year formats, years in the range 1940 to 2039 can be represented. This leads to the possibility of a date overflow condition occurring when converting from a 4-digit-year format to a 2-digit-year format.

The following table lists the formats for date data.

Figure 7-1. Date Formats for Date Data Type

Format name	**Description**	**Format**	**Length**	**Example**
*MDY	Month/Day/Year	mm/dd/yy	8	01/15/91
*DMY	Day/Month/Year	dd/mm/yy	8	15/01/91
*YMD	Year/Month/Day	yy/mm/dd	8	91/01/15
*JUL	Julian	yy/ddd	6	91/015
*ISO	International Standards Organization	yyyy-mm-dd	10	1991-01-15
*USA	IBM USA Standard	mm/dd/yyyy	10	01/15/1991
*EUR	IBM European Standard	dd.mm.yyyy	10	15.01.1991
*JIS	Japanese Industrial Standard Christian Era	yyyy-mm-dd	10	1991-01-15

The following table lists the *LOVAL, *HIVAL, and default values for all the date formats.

Figure 7-2. Date Values

Format name	**Description**	***LOVAL**	***HIVAL**	**Default Value**
*MDY	Month/Day/Year	01/01/40	12/31/39	01/01/01
*DMY	Day/Month/Year	01/01/40	31/12/39	01/01/01
*YMD	Year/Month/Day	40/01/01	39/12/31	01/01/01
*JUL	Julian	40/001	39/365	01/001
*ISO	International Standards Organization	0001-01-01	9999-12-31	0001-01-01
*USA	IBM USA Standard	01/01/0001	12/31/9999	01/01/0001
*EUR	IBM European Standard	01.01.0001	31.12.9999	01.01.0001
*JIS	Japanese Industrial Standard Christian Era	0001-01-01	9999-12-31	0001-01-01

Time Data Type: Time fields have a predetermined size and format They can be defined on the definition specification. Leading and trailing zeros are required for all time data

Time constants or variables used in comparisons or assignments do not have to be in the same format or use the same separators Also, times used for I/O operations such as input fields, output fields or key fields are also converted (if required) to the necessary format for the operation

The default internal format for time variables is *ISO. This default internal format can be overridden globally by the control specification keyword TIMFMT and individually by the definition specification keyword TIMFMT

The hierarchy used when determining the internal time format and separator for a time field is

1 From the TIMFMT keyword specified on the definition specification
2 From the TIMFMT keyword specified on the control specification
3 *ISO

The following table lists the formats for time data

Figure 7-3 Time formats for Time data type

Format name	Description	Format	Length	Example
*HMS	Hours Minutes Seconds	hh mm ss	8	14 00:00
*ISO	International Standards Organization	hh mm ss	8	14 00 00
*USA	IBM USA Standard AM and PM can be any mix of upper and lower case	hh mm AM or hh mm PM	8	02:00 PM
*EUR	IBM European Standard	hh mm ss	8	14 00 00
*JIS	Japanese Industrial Standard Christian Era	hh.mm ss	8	14 00:00

The following table lists the *LOVAL, *HIVAL, and default values for all the time formats

Figure 7-4 Time Values

Format name	Description	*LOVAL	*HIVAL	Default Value
*HMS	Hours Minutes Seconds	00 00 00	24 00 00	00:00:00
*ISO	International Standards Organization	00 00 00	24 00 00	00 00 00
*USA	IBM USA Standard AM and PM can be any mix of upper and lower case.	00 00 AM	12 00 AM	00:00 AM
*EUR	IBM European Standard	00 00 00	24 00 00	00.00.00
*JIS	Japanese Industrial Standard Christian Era	00 00 00	24 00 00	00 00 00

Timestamp Data Type: Timestamp fields have a predetermined size and format They can be defined on the definition specification Timestamp data must be in the format

`yyyy-mm-dd-hh.mm.ss.mmmmmm (length 26).`

Microseconds (.mmmmmm) are optional for timestamp literals and if not provided will be padded on the right with zeros Leading zeros are required for all timestamp data

The default initialization value for a timestamp is midnight of January 1, 0001 (0001-01-01-00 00 00 000000) The *HIVAL value for a timestamp is 9999-12-31-24 00 00 000000 Similarly, the *LOVAL value for timestamp is 0001-01-01-00 00 00 000000

User Date Special Words

The user date special words (UDATE, *DATE, UMONTH, *MONTH, UDAY, *DAY UYEAR, *YEAR) allow the programmer to supply a date for the program at run time The user date special words access the job date that is specified in the job description The user dates can be written out at output time, UDATE and *DATE can be written out using the Y edit code in the format specified by the control specification (For a description of the job date, see the *Work Management* manual, SC41-4306)

Rules for User Date: Remember the following rules when using the user date

- UDATE, when specified in positions 30 through 43 of the output specifications, prints a 6-character numeric date field *DATE, when similarly specified, prints an 8-character (4-digit year portion) numeric date field These special words can be used in three different date formats

  Month/day/year
  Year/month/day
  Day/month/year

  Use the DATEDIT keyword on the control specification to specify the editing to be done If this keyword is not specified, the default is *MDY
- For an interactive program, the user date special words are set when the job starts running For a batch program, they are set when the job is sent to the job queue. In neither case are they updated when the program runs over midnight or when the job date changes. Use the TIME operation code to obtain the time and date while the program is running
- UMONTH, *MONTH, UDAY, *DAY, and UYEAR when specified in positions 30 through 43 of the output specifications, print a 2-position numeric date field *YEAR can be used to print a 4-position numeric date field Use UMONTH or *MONTH to print the month only UDAY or *DAY to print the day only, and UYEAR or *YEAR to print the year only
- UDATE and *DATE can be edited when they are written if the Y edit code is specified in position 44 of the output specifications The DATEDIT(fmt{separator}) keyword on the control specification determines the format and the separator character to be inserted, for example, 12/31/88, 31.12 88 , 12/31/1988
- UMONTH, *MONTH, UDAY, *DAY, UYEAR and *YEAR cannot be edited by the Y edit code in position 44 of the output specifications
- The user date fields cannot be modified This means they cannot be used
  - In the result field of calculations
  - As factor 1 of PARM operations
  - As the factor 2 index of LOOKUP operations
  - With blank after in output specifications
  - As input fields
- The user date special words can be used in factor 1 or factor 2 of the calculation specifications for operation codes that use numeric fields
- User date fields are not date data type fields but are numeric fields

Editing Date Fields

The Y edit code is normally used to edit a 3- to 9-digit date field. It suppresses the leftmost zeros of date fields, up to but not including the digit preceding the first separator. Slashes are inserted to separate the day, month, and year. The DATEDIT(fmt{separator})and ('value') keywords on the control specification can be used to alter edit formats.

Note: The Y edit code is not valid for *YEAR, *MONTH, and *DAY.

The Z edit code removes the sign (plus or minus) from and suppresses the leading zeros of a numeric field. The decimal point is not placed in the field and is not printed.

The Y edit code suppresses the leftmost zeros of date fields, up to but not including the digit preceding the first separator. The Y edit code also inserts slashes (/) between the month, day, and year according to the following pattern.

```
   nn/n
   nn/nn
   nn/nn/n
   nn/nn/nn
  nnn/nn/nn
   nn/nn/nnnn  Format used with M, D or blank in position 19
  nnn/nn/nnnn  Format used with M, D or blank in position 19
 nnnn/nn/nn    Format used with Y in position 19
nnnnn/nn/nn    Format used with Y in position 19
```

Date Operations

Date operations allow you to perform date and time arithmetic, extract portions of a date, time or timestamp field, or test for valid fields. They operate on date, time, and timestamp fields, and character and numeric fields representing dates, times and timestamps. The date operations are:

- ADDDUR (Add Duration)
- EXTRCT (Extract Date/Time/Timestamp)
- SUBDUR (Subtract Duration)
- TEST (Test Date/Time/Timestamp)

With ADDDUR (Add Duration) you can add a duration to a date or time. With SUBDUR (Subtract Duration) you can subtract a duration from a date or time, or calculate the duration between 2 dates, times or timestamps. With EXTRCT (Extract Date/Time/Timestamp) you can extract part of a date, time or timestamp. With TEST (Test Date/Time/Timestamp) you can test for a valid date, time, or timestamp field. The valid duration codes (and their short forms) are:

- *YEARS for the year (*Y)
- *MONTHS for the month (*M)
- *DAYS for the day (*D)
- *HOURS for the hours (*H)
- *MINUTES for the minutes (*MN)
- *SECONDS for the seconds (*S)
- *MSECONDS for the microseconds (*MS).

Adding or Subtracting Dates: When adding (or subtracting) a duration in months to (or from) a date, the general rule is that the month portion is increased (or decreased) by the number of months in the duration, and the day portion is unchanged. The exception to this is when the resulting day portion would exceed the actual number of days in the resulting month. In this case, the resulting day portion is adjusted to the actual month end date

For example, adding one month to '95/05/30' (*YMD format) results in '95/06/30' as expected The resulting month portion has been increased by 1, the day portion is unchanged On the other hand, adding one month to '95/05/31' results in '95/06/30' The resulting month portion has been increased by 1 and the resulting day portion has been adjusted because June has only 30 days

Subtracting one month from '95/03/30' yields '95/02/28' In this case, the resulting month portion is decreased by 1 and the resulting day portion adjusted because February has only 28 days (in non-leap years)

Similar results occur when adding or subtracting a year duration For example adding one year to '92/02/29' results in '93/02/28', an adjusted value since the resulting year is not a leap year

Calculating Durations between Dates: The SUBDUR operation can be used to calculate a duration by subtracting two dates, times, or timestamps The result of the calculation is a complete units, any rounding which is done is downwards The calculation of durations includes microseconds.

For example, if the actual duration is 384 days, and the result is requested in years, the result will be 1 complete year because there are 1 05 years in 384 days A duration of 59 minutes requested in hours will result in 0 hours Here are some additional examples

Duration in	between	and	is
Months	1994-02-28	1994-03-28	1 month
	1994-03-15	1995-03-14	11 months
	1994-03-15	1995-03-15	12 months
Years	1994-03-15	1995-03-14	0 years
	1994-03-31	1995-03-31	1 year
	1970-03-14-23.00.00.000000	1970-03-14-22.00.00.000001	0 years
Hours	1990-03-14-12.34.45.123456	1989-03-14-12.34.45.123457	0 years

Unexpected Results: If adjustment takes place on a date-time addition or subtraction, then a subsequent duration calculation will most likely result in a different duration than the one originally added or subtracted This is because the calculated duration will no longer contain a complete unit, and so, rounding down, will yield a one unit less than expected This is shown in examples 1 and 2 below

A second unexpected result can be seen in examples 3 and 4 Different initial dates give the same result after adding 1 month When subtracting 1 month from the result, it is impossible to arrive at both initial dates

```
1.   '95/05/31' ADDDUR 1:*MONTH    gives '95/06/30'
     '95/06/30' SUBDUR '95/05/31'  gives 0 months

   You might expect the result of the SUBDUR to be 1 month.

2.   '95/06/30' ADDDUR 1:*MONTH    gives '95/07/30'
     '95/07/30' SUBDUR '95/06/30'  gives 1 month

   This is the "expected" result.

3.   '95/01/31' ADDDUR 1:*MONTH    gives '95/02/28'
     '95/01/28' ADDDUR 1:*MONTH    gives '95/02/28'

   Two different dates yield the same date due to adjustment.

     '95/02/28' SUBDUR 1:*MONTH    gives '95/01/28'

   'Reversing' the addition does not result in the original dates.
```

Using OPM RPG/400 Date Support

The OPM RPG 400 compiler has the following support for dates

- Retrieval of the job date via special words UDATE and UYEAR This level of support provides for a 2-digit year
- Retrieval of the job date via special words *DATE and *YEAR This level of support provides for a 4-digit year
- Retrieval of the system date via operation TIME This level of support provides for both 2-digit and 4-digit years
- Calling OPM System APIs This level of support provides for both 2-digit and 4-digit years
- SQL date, time, and timestamp data type support This level of support provides for 4-digit years

Using System/36 Compatible Date Support

The System/36 compatible compiler has the following support for dates

- Retrieval of the job date via special words UDATE and UYEAR This level of support provides for a 2-digit year
- Retrieval of the system date via operation TIME This level of support provides for a 2-digit year
- Calling OPM System APIs This level of support provides for both 2-digit and 4-digit years

The IBM COBOL Family for AS/400

For application code written in COBOL, IBM has developed several COBOL language compilers These compilers target the host application development environment

IBM's COBOL compilers for AS/400 are:

- Integrated Language Environment (ILE) COBOL/400
- Original Program Model (OPM) COBOL/400
- System/36 compatible COBOL compiler

All three COBOL language compilers are available with either the Integrated Language Environment COBOL for OS/400 Version 3 product (5716-CB1) or the IBM Integrated Language Environment COBOL/400 Version 3 product (5763-CB1)

Using COBOL Date Support

The COBOL ACCEPT statement is supported by the three compilers listed above This level of support provides for a 2-digit year when referencing DATE or DAY

Calling OPM System APIs is supported by the three compilers listed above This level of support provides for both 2-digit and 4-digit years

The COBOL WHEN-COMPILED special register is supported in both the OPM and ILE COBOL/400 compilers. This level of support provides for a 2-digit year

Calling ILE Date APIs is supported by the ILE COBOL/400 compiler This level of support provides for both 2-digit and 4-digit years.

SQL is supported by both the OPM and ILE COBOL compilers This level of support provides for 4-digit years

The IBM C Family for AS/400

For application code based on C, IBM has developed language compilers for both C and C++

IBM's C based compilers for AS/400 are.

- Integrated Language Environment (ILE) C for OS/400
- VisualAge C++ for OS/400

Using C and C++ Date Support

The C library, available to both compilers, provides many functions related to date and time retrieval, manipulation, and formatting This level of support provides for 4-digit years

Calling OPM System APIs is supported by both compilers This level of support provides for both 2-digit and 4-digit years

Calling ILE Date APIs is supported by both compilers This level of support provides for both 2-digit and 4-digit years.

SQL is supported by the ILE C for OS/400 compiler. This level of support provides for 4-digit years

Integrated Language Environment for OS/400

Integrated Language Environment (ILE) for OS/400 is IBM's common runtime environment for enterprise applications written in the ILE languages of RPG, COBOL, C, and CL ILE is designed to provide defined calling conventions, enhanced interlanguage communication, and callable services in areas such as date and time, math, storage management, and exception handling ILE services are standard in OS/400 starting with Version 2 Release 3

ILE provides a number of date functions which include.

- Ability to parse dates in an infinite number of formats by using a picture string as a parsing guide
- Retrieve current date
- Convert a date character string to a Lilian (integer) format thereby enabling easy date arithmetic operations
- Convert a Lilian date to a date character string
- Century sliding window

The ILE century sliding window technique may provide a short term solution for some applications. If you are unable to change all of your applications and data at the same time, the century window allows 2-digit years to be interpreted in a 100-year window Any 100-year window can be selected You pass a 2-digit year to ILE and ILE returns a 4-digit year based on the 100-year window

The advantage to the century window is that you need to change only the application code and not the databases with 2-digit years This allows you to change the application programs one at a time or groups at a time without affecting your databases Note, this only works for dates that range less than 100 years For example, dates of birth may not be appropriate for this solution

DB2/400 SQL

DB2/400, a standard part of Operating System/400, provides the run time support for SQL on the AS/400 SQL provides support for the database datatypes of date, time, and timestamp, and operations such as add, subtract, assignment, and compare The DB2 Query Manager and SQL Development Kit product (5716-ST1 or 5763-ST1) provides SQL precompilers for AS/400 programming languages such as RPG, C, and COBOL SQL support provides for 4-digit year support.

Using Date, Time, and Timestamp Support

Date, time, and timestamp are data types represented in an internal form not seen by the SQL user Date, time, and timestamp can be represented by character string values and assigned to character string variables The database manager recognizes the following as date, time, and timestamp values

- A value returned by the DATE, TIME, or TIMESTAMP scalar functions
- A value returned by the CURRENT DATE, CURRENT TIME, or CURRENT TIMESTAMP special registers
- A character string when it is an operand of an arithmetic expression or a comparison *and* the other operand is a date, time, or timestamp For example, in the predicate

```
... WHERE HIREDATE < '1950-01-01'
```

if HIREDATE is a date column, the character string '1950-01-01' is interpreted as a date.

- A character string variable or constant used to set a date, time, or timestamp column in either the SET clause of an UPDATE statement, or the VALUES clause of an INSERT statement

For more information on character string formats of date, time, and timestamp values, refer to *DB2/400 SQL Reference* (SC41-3612).

Specifying Current Date and Time Values: You can specify a current date, time, or timestamp in an expression by specifying one of three special registers CURRENT DATE, CURRENT TIME, or CURRENT TIMESTAMP The value of each is based on a time-of-day clock reading obtained during the running of the statement Multiple references to CURRENT DATE, CURRENT TIME, or CURRENT TIMESTAMP within the same SQL statement use the same value The following statement returns the age (in years) of each employee in the EMPLOYEE table when the statement is run.

```
SELECT YEAR(CURRENT DATE - BIRTHDATE)
FROM CORPDATA.EMPLOYEE
```

The CURRENT TIMEZONE special register allows a local time to be converted to Universal Coordinated Time (UTC) For example, if you have a table named DATETIME, containing a time column type with a name of STARTT, and you want to convert STARTT to UTC, you can use the following statement.

```
SELECT STARTT - CURRENT TIMEZONE
FROM DATETIME
```

Datetime Operands and Durations: Datetime values can be incremented, decremented, and subtracted These operations may involve decimal numbers called *durations* A *duration* is a positive or negative number representing an interval of time There are four types of durations·

Labeled Durations

A *labeled duration* represents a specific unit of time as expressed by a number (which can be the result of an expression) followed by one of the seven duration keywords YEARS, MONTHS, DAYS, HOURS, MINUTES, SECONDS, or MICROSECONDS[1] The number specified is converted as if it were assigned to a DECIMAL(15,0) number A labeled duration can only be used as an operand of an arithmetic operator in which the other operand is a value of data type DATE, TIME, or TIMESTAMP. Thus, the expression HIREDATE + 2 MONTHS + 14 DAYS is valid whereas the expression HIREDATE + (2 MONTHS + 14 DAYS) is not. In both of these expressions, the labeled durations are 2 MONTHS and 14 DAYS.

Date Duration

A *date duration* represents a number of years, months, and days, expressed as a DECIMAL(8,0) number To be properly interpreted, the number must have the format yyyy*mmdd*, where *yyyy* represents the number of years, *mm* the number of months, and *dd* the number of days The result of subtracting one date value from another, as in the expression HIREDATE - BRTHDATE, is a date duration

---

ote that the singular form of these keywords is also acceptable: YEAR, MONTH, DAY, HOUR, MINUTE, SECOND, and ICROSECOND

Time Duration

A *time duration* represents a number of hours, minutes, and seconds, expressed as a DECIMAL(6,0) number. To be properly interpreted, the number must have the format *hhmmss* where *hh* represents the number of hours, *mm* the number of minutes, and ss the number of seconds. The result of subtracting one time value from another is a time duration.

Timestamp duration

A *timestamp duration* represents a number of years, months, days, hours, minutes, seconds, and microseconds, expressed as a DECIMAL(20,6) number. To be properly interpreted, the number must have the format *yyyymmddhhmmsszzzzzz*, where *yyyy*, *mm*, *dd*, *hh*, *mm*, *ss*, and *zzzzzz* represent, respectively, the number of years, months, days, hours, minutes, seconds, and microseconds. The result of subtracting one timestamp value from another is a timestamp duration.

Datetime Arithmetic in SQL: The only arithmetic operations that can be performed on datetime values are addition and subtraction. If a datetime value is the operand of addition, the other operand must be a duration. The specific rules governing the use of the addition operator with datetime values follow.

- If one operand is a date, the other operand must be a date duration or labeled duration of years, months, or days.
- If one operand is a time, the other operand must be a time duration or a labeled duration of hours, minutes, or seconds.
- If one operand is a timestamp, the other operand must be a duration. Any type of duration is valid.
- Neither operand of the addition operator can be a parameter marker.

The rules for the use of the subtraction operator on datetime values are not the same as those for addition because a datetime value cannot be subtracted from a duration, and because the operation of subtracting two datetime values is not the same as the operation of subtracting a duration from a datetime value. The specific rules governing the use of the subtraction operator with datetime values follow.

- If the first operand is a date, the second operand must be a date, a date duration, a string representation of a date, or a labeled duration of years, months, or days.
- If the second operand is a date, the first operand must be a date, or a string representation of a date.
- If the first operand is a time, the second operand must be a time, a time duration, a string representation of a time, or a labeled duration of hours, minutes, or seconds.
- If the second operand is a time, the first operand must be a time, or string representation of a time.
- If the first operand is a timestamp, the second operand must be a timestamp, a string representation of a timestamp, or a duration.
- If the second operand is a timestamp, the first operand must be a timestamp or a string representation of a timestamp.
- Neither operand of the subtraction operator can be a parameter marker.

Date Arithmetic: Dates can be subtracted, incremented, or decremented.

Subtracting Dates The result of subtracting one date (DATE2) from another (DATE1) is a date duration that specifies the number of years, months, and days between the two dates. The data type of the result is DECIMAL(8,0). If DATE1 is greater than or equal to DATE2, DATE2 is subtracted from DATE1. If DATE1 is less than DATE2, however, DATE1 is subtracted from DATE2, and the sign of the result is made negative. The following procedural description clarifies the steps involved in the operation RESULT = DATE1 - DATE2.

```
If DAY(DATE2) <= DAY(DATE1)
    then DAY(RESULT) = DAY(DATE1) - DAY(DATE2)

If DAY(DATE2) > DAY(DATE1)
    then DAY(RESULT) = N + DAY(DATE1) - DAY(DATE2)
      where N = the last day of MONTH(DATE2)
    MONTH(DATE2) is then incremented by 1.

If MONTH(DATE2) <= MONTH(DATE1)
    then MONTH(RESULT) = MONTH(DATE1) - MONTH(DATE2).

If MONTH(DATE2) > MONTH(DATE1)
    then MONTH(RESULT) = 12 + MONTH(DATE1) - MONTH(DATE2).
    YEAR(DATE2) is then incremented by 1.

YEAR(RESULT) = YEAR(DATE1) - YEAR(DATE2).
```

For example, the result of DATE('3/15/2000') - '12/31/1999' is 215 (or, a duration of 0 years, 2 months, and 15 days).

Incrementing and Decrementing Dates The result of adding a duration to a date, or of subtracting a duration from a date, is itself a date. (For the purposes of this operation, a month denotes the equivalent of a calendar page. Adding months to a date, then, is like turning the pages of a calendar, starting with the page on which the date appears.) The result must fall between the dates January 1, 0001 and December 31, 9999 inclusive. If a duration of years is added or subtracted, only the year portion of the date is affected. The month is unchanged, as is the day unless the result would be February 29 of a non-leap-year. In this case, the day is changed to 28, and SQLWARN6 in the SQLCA is set to 'W' to indicate the end-of-month adjustment.

Similarly, if a duration of months is added or subtracted, only months and, if necessary, years are affected. The day portion of the date is unchanged unless the result would be invalid (September 31, for example). In this case, the day is set to the last day of the month, and SQLWARN6 in the SQLCA is set to 'W' to indicate the end-of-month adjustment.

Adding or subtracting a duration of days will, of course, affect the day portion of the date, and potentially the month and year. Adding a labeled duration of DAYS will not cause an end-of-month adjustment.

Date durations, whether positive or negative, may also be added to and subtracted from dates. As with labeled durations, the result is a valid date, and a warning indicator is set in the SQLCA whenever an end-of-month adjustment is necessary.

When a positive date duration is added to a date, or a negative date duration is subtracted from a date, the date is incremented by the specified number of years, months, and days, in that order. Thus DATE1 + X, where X is a positive DECIMAL(8,0) number, is equivalent to the expression

```
DATE1 + YEAR(X) YEARS + MONTH(X) MONTHS + DAY(X) DAYS
```

When a positive date duration is subtracted from a date, or a negative date duration is added to a date, the date is decremented by the specified number of days, months, and years, in that order. Thus, DATE1 - X, where X is a positive DECIMAL(8,0) number, is equivalent to the expression.

```
DATE1 - DAY(X) DAYS - MONTH(X) MONTHS - YEAR(X) YEARS
```

When adding durations to dates, adding one month to a given date gives the same date one month later *unless* that date does not exist in the later month. In that case, the date is set to that of the last day of the later month. For example, January 28 plus one month gives February 28, and one month added to January 29, 30, or 31 results in either February 28 or, for a leap year, February 29.

Note: If one or more months is added to a given date and then the same number of months is subtracted from the result, the final date is not necessarily the same as the original date.

Time Arithmetic: Times can be subtracted, incremented, or decremented.

Subtracting Times. The result of subtracting one time (TIME2) from another (TIME1) is a time duration that specifies the number of hours, minutes, and seconds between the two times. The data type of the result is DECIMAL(6,0). If TIME1 is greater than or equal to TIME2, TIME2 is subtracted from TIME1. If TIME1 is less than TIME2, however, TIME1 is subtracted from TIME2, and the sign of the result is made negative. The following procedural description clarifies the steps involved in the operation RESULT = TIME1 - TIME2.

```
If SECOND(TIME2) < = SECOND(TIME1)
    then SECOND(RESULT) = SECOND(TIME1) - SECOND(TIME2)
If SECOND(TIME2) > SECOND(TIME1)
    then SECOND(RESULT) = 60 + SECOND(TIME1) - SECOND(TIME2)
    MINUTE(TIME2) is then incremented by 1
If MINUTE(TIME2) < = MINUTE(TIME1)
    then MINUTE(RESULT) = MINUTE(TIME1) - MINUTE(TIME2)
If MINUTE(TIME2) > MINUTE(TIME1)
    then MINUTE(RESULT) = 60 + MINUTE(TIME1) - MINUTE(TIME2)
    HOUR(TIME2) is then incremented by 1.
HOUR(RESULT) = HOUR(TIME1) - HOUR(TIME2)
```

For example, the result of TIME('11:02:26') - '00:32.56' is 102930 (a duration of 10 hours, 29 minutes, and 30 seconds).

Incrementing and Decrementing Times: The result of adding a duration to a time, or of subtracting a duration from a time, is itself a time. Any overflow or underflow of hours is discarded, thereby ensuring that the result is always a time. If a duration of hours is added or subtracted, only the hours portion of the time is affected. The minutes and seconds are unchanged.

Similarly, if a duration of minutes is added or subtracted, only minutes and, if necessary, hours are affected. The seconds portion of the time is unchanged.

Adding or subtracting a duration of seconds will, of course, affect the seconds portion of the time, and potentially the minutes and hours.

Time durations, whether positive or negative, also can be added to and subtracted from times. The result is a time that has been incremented or decremented by the specified number of hours, minutes, and seconds, in that order. `TIME1 + X`, where "X" is a DECIMAL(6,0) number, is equivalent to the expression:

```
TIME1 + HOUR(X) HOURS + MINUTE(X) MINUTES + SECOND(X) SECONDS
```

Timestamp Arithmetic: Timestamps can be subtracted, incremented, or decremented.

Subtracting Timestamps: The result of subtracting one timestamp (TS2) from another (TS1) is a timestamp duration that specifies the number of years, months, days, hours, minutes, seconds, and microseconds between the two timestamps. The data type of the result is DECIMAL(20,6). If TS1 is greater than or equal to TS2, TS2 is subtracted from TS1. If TS1 is less than TS2, however, TS1 is subtracted from TS2 and the sign of the result is made negative. The following procedural description clarifies the steps involved in the operation RESULT = TS1 - TS2:

```
If MICROSECOND(TS2) <= MICROSECOND(TS1)
    then MICROSECOND(RESULT) = MICROSECOND(TS1) -
    MICROSECOND(TS2)

If MICROSECOND(TS2) > MICROSECOND(TS1)
    then MICROSECOND(RESULT) = 1000000 +
    MICROSECOND(TS1) - MICROSECOND(TS2)
    and SECOND(TS2) is incremented by 1
```

The seconds and minutes part of the timestamps are subtracted as specified in the rules for subtracting times.

```
If HOUR(TS2) <= HOUR(TS1)
    then HOUR(RESULT) = HOUR(TS1) - HOUR(TS2)

If HOUR(TS2) > HOUR(TS1)
    then HOUR(RESULT) = 24 + HOUR(TS1) - HOUR(TS2)
    and DAY(TS2) is incremented by 1.
```

The date part of the timestamps is subtracted as specified in the rules for subtracting dates.

Incrementing and Decrementing Timestamps. The result of adding a duration to a timestamp, or of subtracting a duration from a timestamp, is itself a timestamp. Date and time arithmetic is performed as previously defined, except that an overflow or underflow of hours is carried into the date part of the result, which must be within the range of valid dates. Microseconds overflow into seconds.

OS/400 CL

CVTDAT (Convert Date)

The Convert Date (CVTDAT) command converts a date value from one format to another, without changing its value. CVTDAT provides for both 2-digit and 4-digit-year formats.

OPNQRYF (Open Query File)

The Open Query File (OPNQRYF) command opens a file that contains a set of database records that satisfies a database query request. OPNQRYF provides for both 2-digit and 4-digit years.

Date, Time, and Timestamp Comparisons Using the OPNQRYF Command: A date, time, or timestamp value can be compared either with another value of the same data type or with a string representation of that data type. All comparisons are chronological, which means the farther a time is from January 1, 0001, the *greater* the value of that time.

Comparisons involving time values and string representations of time values always include seconds. If the string representation omits seconds, zero seconds are implied.

Comparisons involving timestamp values are chronological without regard to representations that might be considered equivalent. Thus, the following predicate is true:

```
TIMESTAMP('1990-02-23-00.00.00') > '1990-02-22-24.00.00'
```

When a character, DBCS-open, or DBCS-either field or constant is represented as a date, time, or timestamp, the following rules apply:

Date: The length of the field or literal must be at least 8 if the date format is *ISO, *USA, *EUR, *JIS, *YMD, *MDY, or *DMY. If the date format is *JUL (yyddd), the length of the variable must be at least 6 (includes the separator between yy and ddd). The field or literal may be padded with blanks.

Time: For all of the time formats (*USA, *ISO, *EUR, *JIS, *HMS), the length of the field or literal must be at least 4. The field or literal may be padded with blanks.

Timestamp: For the timestamp format (yyyy-mm-dd-hh.mm.ss.uuuuuu), the length of the field or literal must be at least 16. The field or literal may be padded with blanks.

Date, Time, and Timestamp Arithmetic Using OPNQRYF CL Command: Date, time, and timestamp values can be incremented, decremented, and subtracted. These operations may involve decimal numbers called *durations*. Following is a definition of durations and a specification of the rules for performing arithmetic operations on date, time, and timestamp values.

Durations: A **duration** is a number representing an interval of time The four types of durations are

Labeled Duration

A **labeled duration** represents a specific unit of time as expressed by a number (which can be the result of an expression) used as an operand for one of the seven duration built-in functions. %DURYEAR, %DURMONTH, %DURDAY, %DURHOUR, %DURMINUTE, %DURSEC, or %DURMICSEC. The functions are for the duration of year, month, day, hour, minute, second, and microsecond, respectively The number specified is converted as if it was assigned to a DECIMAL(15,0) number A labeled duration can only be used as an operand of an arithmetic operator when the other operand is a value of data type *DATE, *TIME, or *TIMESTP Thus, the expression HIREDATE + %DURMONTH(2) + %DURDAY(14) is valid, whereas the expression HIREDATE + (%DURMONTH(2) + %DURDAY(14)) is not In both of these expressions, the labeled durations are %DURMONTH(2) and %DURDAY(14).

Date Duration

A **date duration** represents a number of years, months, and days, expressed as a DECIMAL(8,0) number To be properly interpreted, the number must have the format *yyyymmdd*, where *yyyy* represents the number of years, *mm* the number of months, and *dd* the number of days The result of subtracting one date value from another, as in the expression HIREDATE - BRTHDATE, is a date duration

Time Duration

A **time duration** represents a number of hours, minutes, and seconds, expressed as a DECIMAL(6,0) number. To be properly interpreted, the number must have the format *hhmm*ss, where *hh* represents the number of hours, *mm* the number of minutes, and ss the number of seconds The result of subtracting one time value from another is a time duration

Timestamp Duration

A **timestamp duration** represents a number of years, months, days hours, minutes, seconds, and microseconds, expressed as a DECIMAL(20,6) number To be properly interpreted, the number must have the format *yyyymmddhhmmsszzzzzz*, where *yyyy*, *mm*, *dd*, *hh*, *mm*, ss, and *zzzzzz* represent, respectively, the number of years, months, days, hours, minutes, seconds, and microseconds The result of subtracting one timestamp value from another is a timestamp duration

Rules for Date, Time, and Timestamp Arithmetic: The only arithmetic operations that can be performed on date and time values are addition and subtraction If a date or time value is the operand of addition, the other operand must be a duration. The specific rules governing the use of the addition operator with date and time values follow

- If one operand is a date, the other operand must be a date duration or a labeled duration of years, months, or days
- If one operand is a time, the other operand must be a time duration or a labeled duration of hours, minutes, or seconds

- If one operand is a timestamp, the other operand must be a duration. Any type of duration is valid.

The rules for the use of the subtraction operator on date and time values are not the same as those for addition because a date or time value cannot be subtracted from a duration, and because the operation of subtracting two date and time values is not the same as the operation of subtracting a duration from a date or time value. The specific rules governing the use of the subtraction operator with date and time values follow.

- If the first operand is a date, the second operand must be a date, a date duration, a string representation of a date, or a labeled duration of years, months, or days.
- If the second operand is a date, the first operand must be a date or a string representation of a date.
- If the first operand is a time, the second operand must be a time, a time duration, a string representation of a time, or a labeled duration of hours, minutes, or seconds.
- If the second operand is a time, the first operand must be a time or string representation of a time.
- If the first operand is a timestamp, the second operand must be a timestamp, a string representation of a timestamp, or a duration.
- If the second operand is a timestamp, the first operand must be a timestamp or a string representation of a timestamp.

Date Arithmetic: Dates can be subtracted, incremented, or decremented.

Subtracting Dates: The result of subtracting one date (DATE2) from another (DATE1) is a date duration that specifies the number of years, months, and days between the two dates. The data type of the result is DECIMAL(8,0). If DATE1 is greater than or equal to DATE2, DATE2 is subtracted from DATE1. If DATE1 is less than DATE2, however, DATE1 is subtracted from DATE2, and the sign of the result is made negative. The following procedural description clarifies the steps involved in the operation RESULT = DATE1 - DATE2.

```
If %DAY(DATE2) < = %DAY(DATE1)
    then %DAY(RESULT) = %DAY(DATE1) - %DAY(DATE2)
If %DAY(DATE2) > %DAY(DATE1)
    then %DAY(RESULT) = N + %DAY(DATE1) - %DAY(DATE2)
     where N = the last day of %MONTH(DATE2).
    %MONTH(DATE2) is then incremented by 1
If %MONTH(DATE2) < = %MONTH(DATE1)
    then %MONTH(RESULT) = %MONTH(DATE1) - %MONTH(DATE2).
If %MONTH(DATE2) > %MONTH(DATE1)
    then %MONTH(RESULT) = 12 + %MONTH(DATE1) - %MONTH(DATE2)
    %YEAR(DATE2) is then incremented by 1
%YEAR(RESULT) = %YEAR(DATE1) - %YEAR(DATE2)
```

For example, the result of %DATE('3/15/2000') - '12/31/1999' is 215 (or, a duration of 0 years, 2 months, and 15 days).

Incrementing and Decrementing Dates. The result of adding a duration to a date, or of subtracting a duration from a date, is itself a date (For the purposes of this operation, a month denotes the equivalent of a calendar page Adding months to a date, then, is like turning the pages of a calendar, starting with the page on which the date appears.) The result must fall between the dates January 1, 0001 and December 31, 9999, inclusive. If a duration of years is added or subtracted only the year portion of the date is affected The month is unchanged as is the day unless the result would be February 29 of a year that is not a leap year In this case, the day is changed to 28

Similarly, if a duration of months is added or subtracted, only months and, if necessary, years are affected The day portion of the date is unchanged unless the result would not be valid (September 31, for example). In this case, the day is set to the last day of the month.

Adding or subtracting a duration of days will, of course, affect the day portion of the date, and potentially the month and year

Date durations, whether positive or negative, may also be added to and subtracted from dates As with labeled durations, the result is a valid date.

When a positive date duration is added to a date, or a negative date duration is subtracted from a date, the date is incremented by the specified number of years, months, and days, in that order Thus, DATE1 + X, where X is a positive DECIMAL(8,0) number, is equivalent to the expression. DATE1 + %DURYEAR(%YEAR(X)) + %DURMONTH(%MONTH(X)) + %DURDAY(%DAY(X))

When a positive date duration is subtracted from a date, or a negative date duration is added to a date, the date is decremented by the specified number of days, months, and years, in that order Thus, DATE1 - X, where X is a positive DECIMAL(8,0) number, is equivalent to the expression DATE1 - %DURDAY(%DAY(X)) - %DURMONTH(%MONTH(X)) - %DURYEAR(%YEAR(X))

When adding durations to dates, adding one month to a given date gives the same date one month later *unless* that date does not exist in the later month In that case, the date is set to that of the last day of the later month For example, January 28 plus one month gives February 28, and one month added to January 29, 30, or 31 results in either February 28 or, for a leap year, February 29

Note: If one or more months are added to a given date and then the same number of months is subtracted from the result, the final date is not necessarily the same as the original date

Time Arithmetic: Times can be subtracted, incremented, or decremented

Subtracting Times. The result of subtracting one time (TIME2) from another (TIME1) is a time duration that specifies the number of hours, minutes, and seconds between the two times The data type of the result is DECIMAL(6,0) If TIME1 is greater than or equal to TIME2, TIME2 is subtracted from TIME1 If TIME1 is less than TIME2, however, TIME1 is subtracted from TIME2, and the sign of the result is made negative The following procedural description clarifies the steps involved in the operation RESULT = TIME1 - TIME2

```
If %SECOND(TIME2) <= %SECOND(TIME1)
    then %SECOND(RESULT) = %SECOND(TIME1) - %SECOND(TIME2)
If %SECOND(TIME2) > %SECOND(TIME1)
    then %SECOND(RESULT) = 60 + %SECOND(TIME1) - %SECOND(TIME2)
    %MINUTE(TIME2) is then incremented by 1
If %MINUTE(TIME2) <= %MINUTE(TIME1)
    then %MINUTE(RESULT) = %MINUTE(TIME1) - %MINUTE(TIME2)
If %MINUTE(TIME2) > %MINUTE(TIME1)
    then %MINUTE(RESULT) = 60 + %MINUTE(TIME1) - %MINUTE(TIME2)
    %HOUR(TIME2) is then incremented by 1
%HOUR(RESULT) = %HOUR(TIME1) - %HOUR(TIME2)
```

For example, the result of %TIME('11.02.26') - '00 32 56' is 102930 (a duration of 10 hours, 29 minutes, and 30 seconds)

Incrementing and Decrementing Times The result of adding a duration to a time, or of subtracting a duration from a time, is itself a time Any overflow or underflow of hours is discarded, thereby ensuring that the result is always a time If a duration of hours is added or subtracted, only the hours portion of the time is affected The minutes and seconds are unchanged

Similarly, if a duration of minutes is added or subtracted, only minutes and, if necessary, hours are affected The seconds portion of the time is unchanged

Adding or subtracting a duration of seconds will, of course, affect the seconds portion of the time, and potentially the minutes and hours

Time durations, whether positive or negative, also can be added to and subtracted from times The result is a time that has been incremented or decremented by the specified number of hours, minutes, and seconds, in that order `TIME1 + X`, where `X` is a DECIMAL(6,0) number, is equivalent to the expression TIME1 + %DURHOUR(%HOUR(X)) + %DURMINUTE(%MINUTE(X)) + %DURSEC(%SECOND(X))

Timestamp Arithmetic: Timestamps can be subtracted, incremented, or decremented

Subtracting Timestamps The result of subtracting one timestamp (TS2) from another (TS1) is a timestamp duration that specifies the number of years, months, days hours, minutes, seconds, and microseconds between the two timestamps The data type of the result is DECIMAL(20,6) If TS1 is greater than or equal to TS2, TS2 is subtracted from TS1 If TS1 is less than TS2, however, TS1 is subtracted from TS2 and the sign of the result is made negative The following procedural description clarifies the steps involved in the operation RESULT = TS1 - TS2

```
If %MICSEC(TS2) <= %MICSEC(TS1)
    then %MICSEC(RESULT) = %MICSEC(TS1) - %MICSEC(TS2)
If %MICSEC(TS2) > %MICSEC(TS1)
    then %MICSEC(RESULT) = 1000000 + %MICSEC(TS1) - %MICSEC(TS2)
    and %SECOND(TS2) is incremented by 1
```

The seconds and minutes part of the timestamps are subtracted as specified in the rules for subtracting times

```
If %HOUR(TS2) < = %HOUR(TS1)
    then %HOUR(RESULT) = %HOUR(TS1) - %HOUR(TS2)
If %HOUR(TS2) > %HOUR(TS1)
    then %HOUR(RESULT) = 24 + %HOUR(TS1) - %HOUR(TS2)
    and %DAY(TS2) is incremented by 1
```

The date part of the timestamp is subtracted as specified in the rules for subtracting dates

Incrementing and Decrementing Timestamps The result of adding a duration to a timestamp, or of subtracting a duration from a timestamp, is itself a timestamp Date and time arithmetic is performed as previously defined, except that an overflow or underflow of hours is carried into the date part of the result, which must be within the range of valid dates Microseconds overflow into seconds

pplication Dictionary Services/400

IBM Application Dictionary Services/400 is a host-based, integrated impact analysis tool to improve programmer productivity and application quality during application development and maintenance

The product is a feature of the Application Development ToolSet/400 and is integrated with such tools as Source Entry Utility (SEU), Screen Design Aid (SDA), and Data File Utility (DFU) It is fully menu-driven, very simple to learn and has virtually no learning curve for experienced AS/400 programmers

Application Dictionary Services provides the ability to place a "dictionary" or a cross-referencing index over data and programs When a programmer wants to make a change to an application program, he or she uses the dictionary to determine the effect that change will have on other programs, files, and data The programmer is saved from manually performing the tedious and error-prone work of looking for these relationships This, coupled with the Application Dictionary Services mass compile feature, can improve the quality of applications and decrease the amount of time programmers spend maintaining existing code

With the pervasiveness of date information in many applications, this impact analysis and mass compile capability can simplify your transition to a Year2000-ready application base

pplication Development Manager/400

IBM Application Development Manager/400 is a host-based change management tool that provides application developers a development environment to effectively and efficiently manage application development and maintenance

The product is integrated with the AS/400 Product Development Manager (PDM) and is a feature of Application Development ToolSet/400 It can be used either from the PDM menu, or by typing the Control Language (CL) commands on the command line

Application Development Manager provides a discipline to better control the development and maintenance environment It forces the project leader to define the application environment, outlining the different stages of application (production, test, or fix) and the roles of each developer. The developers work in a well-organized environment, which in turn, leads to increased productivity. Through its version control facility, it allows the creating and managing of several versions of the same applications with less disk utilization and the audit trail feature keeps a log of all changes made to an application, allowing better monitoring of the application.

With the pervasiveness of date information in many applications, this change management tool can simplify your transition to a Year2000 safe application base.

IBM Tools for Personal Computers

The IBM COBOL Family for the Workstations

For application code written in COBOL for the workstation or developed for the MVS, VM, VSE, or AS/400 platforms, IBM has developed COBOL compilers for the workstation. These products target the workstation and host application development environments.

IBM's COBOL workstation products are:

- **IBM VisualAge for COBOL for OS/2**
- **IBM COBOL Set for AIX**

Features

These COBOL compilers, IBM VisualAge for COBOL for OS/2 and IBM COBOL Set for AIX, are available today and provide full Year2000 support. They provide ANSI COBOL Standard Intrinsic Functions which give full date manipulation capability with 4-digit-year support.

Companies need productive application development environments. A language compiler is only a portion of what application programmers need today to develop and maintain code. Tools that generate statistics from code, modularize, migrate, or restructure code, or search class libraries, are becoming more and more important.

IBM VisualAge for COBOL for OS/2

IBM VisualAge for COBOL for OS/2 provides the COBOL programmer with 32-bit Direct-to-SOM based, object-oriented support on the OS/2 operating system. In addition, a COBOL application development environment is provided that is designed specifically to handle client/server, mission-critical, line-of-business applications through visual development. IBM VisualAge for COBOL for OS/2 also gives the COBOL programmer a set of high-productivity, OS/2-based power tools for the development of applications targeting OS/2 execution systems.

Local and Remote Data Access: IBM VisualAge for COBOL for OS/2 provides local and remote data access to data including:

- IBM SMARTdata UTILITIES (SdU) which provides
  - Record oriented file access through standard COBOL I/O statements to
    - local OS/2 VSAM files
    - remote MVS VSAM, SAM, PDS, and PDSE files
    - remote OS/400 Record Files
    - remote CICS managed VSAM files on MVS using CICS/DDM
  - A full set of data conversion API's for converting single, double and mixed byte character strings, numerics and complex structured records
  - A full set of SMARTsort API's for sorting, copying, and merging record and byte files located locally or remotely
- Direct access to data managed by BTRIEVE
- Support for local and remote DB2 data access using DB2 for OS/2
- Support for local and remote CICS data access using CICS for OS/2 or CICS Client for OS/2

Visual GUI Designer: The visual GUI designer provides capabilities to allow the building of complex CUA-compliant screens. The visual interface (GUI work screen) that creates the GUI code, is an easy-to-use, intuitive tool for creating graphical interfaces, eliminating the need for in-depth GUI development knowledge Programmers can create applications by selecting controls from the control palette, moving them onto the design editor, thereby providing an integrated "what you see is what you get" (WYSIWYG) user interface. Either during or after this brief development process, developers may build the application by coding in COBOL logic with COBOL sensitive edit/compile/debug tools

WorkFrame/2: IBM WorkFrame/2 provides seamless integration of all the components included in the IBM VisualAge for COBOL for OS/2 product WorkFrame/2 is a highly configurable, project-oriented application development environment for use on OS/2 and is specifically designed to take full advantage of the features offered by OS/2 When used as the integration medium for application development tools, the fully configurable IBM WorkFrame/2 increases the effectiveness of these tools as agents for enhancing user productivity IBM WorkFrame/2 organizes the programmer's workplace by grouping files into logical units or projects As an organizer, IBM WorkFrame/2

- Adapts to the user's project organization environment instead of the project organization having to fit into the WorkFrame defined environment
- Sets up projects to consist of source and object files spanning multiple directories, and one target directory, (such as, EXE or DLL).
- Associates each project with multiple actions, including compiling, debugging, making, linking, browsing, profiling/analyzing, and preprocessing
- As a tools integrator, multiple developers can now work concurrently on a single project by plugging in their own source control system.

Context Sensitive Editor: The LPEX Editor is a language-sensitive, color editor which supports COBOL. The LPEX Editor can be used to create and edit many types of text files, including program source and documentation. By automatically parsing COBOL source code, LPEX distinguishes between language constructs For instance, language keywords, comments, string literals, and numbers are displayed using distinctive fonts and colors Developers can quickly find items they are looking for in their source code Using LPEX, developers can

- Be made aware of some syntax errors when the source code is created
- Use multiple windows to display several documents or to display more than one view of the same document
- Dynamically configure LPEX to be a multiple-window or single-window tool
- Select a block of text and move or copy it between documents
- Cut and paste to a shell or another application
- Undo previous changes to a document

Developers can customize and extend virtually every aspect of this programmable editor LPEX is designed to be extended through dynamic link libraries; there is no proprietary extension language to learn With the LPEX application programming interface (API), developers can write powerful extensions to the editor using C and C++. In addition, LPEX provides a rich command language that developers can use to create or modify editor functions Developers can.

- Define their own fonts and colors
- Modify the editor action key layout
- Add menus to perform frequently used commands (menu definitions can be applied on a filename extension basis)

- Write their own editor commands

Interactive Debug Tool for OS/2 (IDbug): The debug tool supplied with IBM VisualAge for COBOL for OS/2 provides source level debugging built around a set of core functions designed to let users quickly and efficiently control execution, and analyze data Users can

- Display and change variables
- Display and change storage
- Display and change the processor registers
- Display the call stack
- Add and delete simple and complex breakpoints
- Control the execution of multiple threads
- View source code as a listing, disassembly or mixed

CICS for OS/2 Version 3 0 transactions built with IBM VisualAge for COBOL for OS/2 can be debugged interactively

For PM application programming support synchronous and asynchronous modes gives users two ways to debug PM applications. The application windows can be managed concurrently with the debug tool windows.

Performance Tuning: Execution trace analysis and performance tuning is provided through the IBM Performance Analyzer. It is designed to help users tune and understand their programs by monitoring program execution and generating a function-by-function trace of the run This trace can subsequently be examined by utility programs that graphically display the execution trace Not only does the analyzer trace procedures in the EXE file, but it traces the entry points to system calls and application DLLs

Data Assistant and Transaction Assistant: Data Assistant simplifies the process of constructing syntactically correct, embedded SQL statements. It gives you a graphical view of your relational database, allows you to map COBOL data structures to the database and generate SQL statements into your source file Transaction Assistant enables non-CICS COBOL applications to access CICS transactions

Product Positioning: IBM's object-oriented COBOL family of products combine the technical and practical advantages of the COBOL language with the benefits of object-oriented programming The fact that these products extend the COBOL language, the world's most popular conventional programming language, gives them four distinct advantages over other object-oriented programming languages

1. The ability to readily interface to, and extend, existing COBOL applications
2. An abundance of developers with COBOL programming skills (which increases the likelihood that object-oriented COBOL will become the most widely-used object-oriented programming language in the industry)
3. The natural synergy of COBOL, transaction processing and database access via object-oriented programming, provides an compelling reason to use IBM's object-oriented COBOL offerings.
4. The object-oriented features are a natural extension of COBOL. It now will be easy for COBOL developers to get started with object-oriented COBOL while using the same robust COBOL development environment they have grown to depend on

Object-oriented COBOL applications can work with existing COBOL applications, as well as SOM objects written in other languages

IBM VisualAge for COBOL for OS/2 works with TeamConnection to store COBOL source files, build COBOL applications, provides version control and change management for applications in production

Year2000 Support: Most applications were coded with 2-digit-year data IBM VisualAge for COBOL for OS/2 provides intrinsic functions not available in our earlier COBOL compilers Calendar functions include

- CURRENT-DATE
- DATE-OF-INTEGER
- DAY-OF-INTEGER
- INTEGER-OF-DATE
- INTEGER-OF-DAY
- WHEN-COMPILED

Example of obtaining the 4-digit year with the COBOL Intrinsic Functions

- DATE-OF-INTEGER gives YYYYMMDD
- DAY-OF-INTEGER gives YYYYDDD

IBM COBOL Set for AIX

IBM COBOL Set for AIX provides the COBOL programmer with Direct-to-SOM-based, object-oriented support on the AIX operating system IBM COBOL Set for AIX provides a COBOL application development environment that is designed specifically to create client/server, mission critical, line-of-business applications IBM COBOL Set for AIX also gives the COBOL programmer a set of high-productivity, AIX-based tools for the development of applications targeting AIX execution systems

IBM COBOL Set for AIX includes local and remote data access, a context sensitive editor (LPEX), Program Builder, Program Debug Tool, Common Desktop Integration of these tools, and COBOL online documentation

Local and Remote Data Access: IBM COBOL Set for AIX provides local and remote data access to data including

- IBM SMARTdata UTILITIES (SdU) which provides
  - Record oriented file access through standard COBOL I/O statements to
    - local AIX VSAM files
    - remote MVS VSAM, SAM, PDS, and PDSE files
    - remote OS/400 files
    - remote CICS managed VSAM files on MVS using CICS/DDM
    - data managed by IBM ENCINA for AIX Shared File System
- Support for local and remote DB2 data access using DB2 for AIX
- Support for local and remote CICS data access using CICS for AIX or CICS Client for AIX

Context Sensitive Editor: The LPEX Editor is a language-sensitive, color editor which supports COBOL The LPEX Editor can be used to create and edit many types of text files, including program source and documentation By automatically parsing COBOL source code, LPEX distinguishes between language constructs For instance, language keywords, comments, string literals, and numbers are displayed using distinctive fonts and colors

Developers can quickly find items they are looking for in their source code Using LPEX, developers can

- Be made aware of some syntax errors when the source code is created
- Use multiple windows to display several documents or to display more than one view of the same document
- Dynamically configure LPEX to be a multiple-window or single-window tool
- Select a block of text and move or copy it between documents
- Cut and paste to a shell or another application
- Undo previous changes to a document

Developers can customize and extend virtually every aspect of this programmable editor LPEX is designed to be extended through shared libraries, there is no proprietary extension language to learn With the LPEX application programming interface (API), developers can write powerful extensions to the editor using C and C++ In addition, LPEX provides a rich command language that developers can use to create or modify editor functions Developers can

- Define their own fonts and colors
- Modify the editor action key layout
- Add menus to perform frequently used commands (menu definitions can be applied on a filename extension basis)
- Write their own editor commands

Program Builder: The Program Builder manages the repetitive tasks of compiling, linking, and correcting errors in program source code The Program Builder

- Provides a graphical user interface to simplify the process of setting and saving compile and linker options
- Lists build errors in a window Selecting a compile error in the list will position you at the error in the source code in the LPEX Editor
- Creates a makefile that is used by the AIX make command to construct and maintain programs and libraries The Program Builder also determines build dependencies by scanning the source code files for dependency information

Common Desktop Environment (CDE): End-user productivity is enhanced on AIX Version 4 with a new user interface for the AIXwindows (R) Desktop which is based on the Common Desktop Environment This new graphical user interface is included on both the AIX for Clients and the AIX for Servers packages

The Common Desktop Environment (CDE) integration for COBOL Set for AIX consists of a COBOL application folder which is integrated within the CDE Application Manager The COBOL Set for AIX application folder contains icons representing the COBOL tools and applications The COBOL application folder will contain icons for the LPEX editor, the Program Builder, the Program Debug Tool, and COBOL Online Documentation

CDE Integration of the COBOL tools will allow the user to invoke the tools in a simple and consistent manner The CDE desktop recognizes different types of files using a data type database A data type identifies the files of a particular format and associates them with the appropriate applications These associations mean that users don't have to remember command line invocations of tools In most cases when a user double-clicks on a file, the CDE desktop will automatically launch the correct application that understands that file's data

Program Debug Tool: The COBOL Set for AIX debug tool helps you detect and diagnose errors in code developed using the COBOL Set for AIX compiler This intuitive graphical user interface allows you to control execution of the program, examine and modify data (variables, storage, and registers), and perform many other useful functions. Additionally, you can debug C functions that your applications may be using

The debug tool provides source-level debugging and is built around a set of core functions designed to let developers quickly and efficiently control execution and analyze data With these core functions, developers can

- Display and change variables
- Display and change storage
- Display and change the processor registers
- Display the call stack
- Add and delete simple and complex breakpoints
- Control the execution of multiple threads
- View source code as listing, disassembly or mixed

Product Positioning: IBM's object-oriented COBOL family of products combine the technical and practical advantages of the COBOL language with the benefits of object-oriented programming The fact that these products extend the COBOL language, the world's most popular conventional programming language, gives them four distinct advantages over other object-oriented programming languages

1 The ability to readily interface to, and extend, existing COBOL applications

2 An abundance of developers with COBOL programming skills (which increases the likelihood that object-oriented COBOL will become the most widely-used object-oriented programming language in the industry)

3 The natural synergy of COBOL, transaction processing and database access via object-oriented programming, and provides a compelling reason to use IBM's object-oriented COBOL offerings

4 The object-oriented features are a natural extension of COBOL It now will be easy for COBOL developers to get started with object-oriented COBOL while using the same robust COBOL development environment they have grown to depend on.

Object-oriented COBOL applications can work with existing COBOL applications, as well as SOM objects written in other languages

Year2000 Support: Most applications were coded with 2-digit-year data IBM COBOL Set for AIX provides intrinsic functions not available in our earlier COBOL compilers Calendar functions include.

- CURRENT-DATE
- DATE-OF-INTEGER
- DAY-OF-INTEGER
- INTEGER-OF-DATE
- INTEGER-OF-DAY
- WHEN-COMPILED

Example of obtaining the 4-digit year with the COBOL Intrinsic Functions:

- DATE-OF-INTEGER gives YYYYMMDD
- DAY-OF-INTEGER gives YYYYDDD

The IBM PL/I Family for the Workstations

For application code written in PL/I for the workstation or developed for the MVS, VM, VSE, or AS/400 platforms, IBM has developed PL/I compilers for the workstation. These products target the workstation and host application development environments

IBM's PL/I workstation products are

- **IBM PL/I for OS/2**
- **IBM PL/I Set for AIX**

Features

These PL/I compilers, IBM PL/I for OS/2 and IBM PL/I Set for AIX, are available today and provide full Year2000 support They provide a Built-in Function which provides 4-digit-year support

Companies need productive application development environments A language compiler is only a portion of what application programmers need today to develop and maintain code. Tools that generate statistics from code, modularize, migrate, or restructure code, or search class libraries, are becoming more and more important.

IBM PL/I for OS/2

IBM PL/I for OS/2 (10H7848, 10H7819, 1322966, 5622-088) provides the PL/I programmer with 32-bit support on the OS/2 operating system In addition, a PL/I application development environment is provided that is designed especially to handle mission critical, line-of-business applications through visual programming and construction-from-components technologies PL/I for OS/2 is offered in three options - a Personal Edition, a Professional Edition, and a Toolkit

IBM PL/I for OS/2 Personal Edition: IBM PL/I for OS/2 Personal Edition was designed for small software development companies, consultants, and students The Personal Edition contains a full 32-bit compiler, run-time library, and graphical, interactive debugging facility that supports new PL/I application development on stand-alone PCs or small LANs

IBM PL/I for OS/2 Professional Edition: IBM PL/I for OS/2 Professional Edition includes all of the features of the Personal Edition, plus additional function that enhances compatibility with the mainframe compiler. Developers can build and test host-based VSAM, DB2 (R), CICS, and IMS applications, or they can take advantage of PL/I for OS/2 (in combination with DB2/2, CICS OS/2, and IMS Client Server/2) to create client/server applications that integrate with existing mainframe programs

IBM PL/I for OS/2 Toolkit: IBM PL/I for OS/2 Toolkit helps you streamline the programming process, offering a development environment for PL/I for OS/2 presentation manager (PM) applications. The Toolkit is a collection of tools designed to complement either the Professional or Personal Edition of PL/I for OS/2 The Toolkit includes a visual prototyping tool and code generator that can help you develop graphical user interfaces, a programming aid designed to help you convert C header files to PL/I header files, and the OS/2 Developer's Toolkit.

IBM PL/I Set for AIX

IBM PL/I Set for AIX (33H1858, 33H5425, 5765-549) provides a PL/I application development environment designed to allow you to create mission critical, line-of-business applications that can run on host systems, workstations, or client/server systems with access to DB2(R), CICS, VSAM/SAM, and other data systems. IBM PL/I Set for AIX provides the PL/I programmer with an optimizing compiler and a set of high-productivity, AIX-based tools integrated with the AIX Common Desktop Environment, for the development of applications.

PL/I Compiler: IBM PL/I Set for AIX provides an optimizing compiler that contains a rich implementation of the PL/I language as well as support to improve compatibility with mainframe PL/I and enhancements that can allow new AIX-based applications to take advantage of features of the AIX platform.

The PL/I compiler also includes powerful, integrated preprocessors. You can select from one or more of the preprocessors as required for use in your program. The preprocessors included are:

- the macro facility
- the include preprocessor that allows you to incorporate external source files
- the SQL preprocessor that translates embedded SQL statements into PL/I statements, providing support for local and remote DB2 data access when used with DB2 for AIX
- the CICS preprocessor that translates embedded CICS statements into PL/I statements, providing support for local and remote CICS data access when used with CICS for AIX or CICS Client for AIX

A choice of linkages and parameter-passing mechanisms is also provided to facilitate interlanguage communications (ILC) between your PL/I routines and C/C++, Fortran, Pascal, and COBOL routines on AIX.

PL/I Set for AIX supports a single-byte character set (SBCS) and a double-byte character set (DBCS).

Common Desktop Environment: IBM PL/I Set for AIX utilizes the new graphical user interface (GUI), based on the Common Desktop Environment (CDE), in IBM AIX Version 4.1. The CDE integration consists of a PL/I application folder which is integrated within the CDE Application Manager. The PL/I Set for AIX application folder contains icons representing the PL/I tools and applications. CDE Integration of the PL/I tools will allow the user to invoke the tools in a simple and consistent manner. The CDE desktop recognizes different types of files using a data type database. A data type identifies the files of a particular format and associates them with the appropriate applications. These associations mean that users don't have to remember command line invocations of tools. In most cases when a user double-clicks on a file, the CDE desktop will automatically launch the correct application that understands that file's data.

The PL/I application folder contains:

- Live Parsing Extensible (LPEX) editor
- Program Builder
- Debug Tool
- PL/I online documentation

LPEX Editor: The LPEX editor is a language-sensitive editor which supports PL/I. The LPEX editor can be used to create and edit many types of text files, including program source and documentation. Using LPEX, developers can:

- Use multiple windows to display several documents or to display more than one view of the same document
- Dynamically configure LPEX to be a multiple-window or single-window tool
- Select a block of text and move or copy it between documents
- Cut and paste to a shell or another application
- Undo previous changes to a document

Developers can customize and extend virtually every aspect of this programmable editor LPEX is designed to be extended through dynamic link libraries. There is no proprietary extension language to learn. With the LPEX application programming interface (API), developers can write powerful extensions to the editor. In addition, LPEX provides a rich command language that developers can use to create or modify editor functions Developers can:

- Define their own fonts and colors
- Modify the editor action key layout
- Add menus to perform frequently used commands (menu definitions can be applied on a filename extension basis)
- Write their own editor commands

Program Builder: The Program Builder manages the repetitive tasks of compiling, linking, and correcting errors in program source code The Program Builder is designed to:

- Provide a graphical user interface to simplify the process of setting and saving compile and linker options
- Lists build errors in a window Selecting a compile error in the list will position you at the error in the source code in the LPEX Editor.
- Creates a makefile that is used by the AIX make command to construct and maintain programs and libraries The Program Builder also determines build dependencies by scanning the source code files for dependency information

Debug Tool: The debug tool helps you detect and diagnose errors in code developed using the PL/I Set for AIX compiler The intuitive graphical user interface allows you to control execution of the program, examine and modify data (variables, storage, and registers), and perform many other useful functions

The debug tool provides machine-level and source-level debugging. It is built around a set of core functions designed to let developers quickly and efficiently control execution, and analyze data With these core functions, developers can

- Display and change variables
- Display and change storage
- Display and change the processor registers
- Display the call stack
- Add and delete simple and complex breakpoints
- Control the execution of multiple threads
- View source code as listing, disassembly or mixed

CICS for AIX Version 2 1 transactions built with IBM PL/I Set for AIX can be debugged interactively

Local and Remote Data Access: IBM PL/I Set for AIX provides the ability to write applications that support local and remote access to data including

- The IBM SMARTdata UTILITIES (SdU) which are designed to provide record oriented file access through standard PL/I I/O statements to
  - local AIX VSAM files

  - remote MVS VSAM, SAM, PDS, and PDSE files
  - remote OS/400 files
  - remote CICS managed VSAM files on MVS through CICS/DDM
- Support for local and remote DB2 data access using DB2 for AIX
- Support for local and remote CICS data access using CICS for AIX or CICS Client for AIX

Solution Developer Tools

Figure 7-5 lists Solution Developer analysis/re-engineering/development tools that aid in reformatting the year-date notation.

Disclaimer

The following is a partial listing of some Solution Developers who represented to us that they provide tools to assist in achieving Year2000 readiness. IBM does not make any representations, endorsement, or warranties of any of the following products. The following information was provided by the Solution Developer and no effort has been made by IBM to independently verify the accuracy of the information relating to the Year2000 date transition. The reader is responsible for checking with the individual Solution Developer to confirm the specific implementation of the Year2000 date transition.

If your company markets a tool, and you would like IBM to consider listing that tool here, please complete and return (by mail or FAX) the "Year2000-Ready Solution Developer Product and Tool Authorization" form provided at the back of this book.

Solution Developer Contacts and Product Name

Figure 7-5 (Page 1 of 4) Solution Developer Tools

Company Name	Contact	Product Name
ADPAC Corp.	425 Market Street, 4th Floor San Francisco, CA 94105 Tel: 415-777-5400 Fax: 415-546-7130	SystemVision YEAR 2000
COGNICASE	425 Viger Ouest Suite 303 Montreal, Quebec H2Z 1X2 Canada Tel: 800-322-3386 Tel. 514-866-6161 Fax: 514-866-6260	Cogni-2000
Computer Software Corp.	19100 Detroit Road Cleveland, OH 44116 Tel: 800-908-2000 Tel. 216-333-4420 Fax 216-333-8288	DateServer 2000
Compuware Corp.	31440 Northwestern Highway Farmington Hills, Michigan 48334 Tel: 800-535-8707 Tel: 810-737-7300 Fax 810-737-2718	PATHVU RETROFIT XPEDITER/TSO XPEDITER/CICS XPEDITER+ XPEDITER/Xchange
EDGE Information Group	2250 East Devon Des Plaines, IL 60018-4510 Tel: 513-948-8906	Edge Portfolio Analyzer

Figure 7-5 (Page 2 of 4) Solution Developer Tools

Company Name	**Contact**	**Product Name**
Global Software, Inc.	P.O. Box 2813 15 Depot Street Duxbury, MA 02331-2813 Tel: 617-934-0949 Fax: 617-934-2001 e-mail: GILES@globsoft.com	GILES
Informission Group, Inc.	7811 L. H. LaFontaine Blvd., Suite 200 Anjou, Quebec H1K 4E4 Canada Tel: 514-351-7755 Fax. 514-351-5845 e-mail: RECY2000@INFORMISSION.CA	RECY2000
Ironsoft, Inc.	4323 Winnequah Rd Monona, WI 53716 Tel: 800-236-0141 Tel: 608-695-1896 Fax: 608-221-8018	Analyzer 2000
ISOGON Corp.	330 Seventh Avenue New York, New York 10001 Tel: 800-568-8828 Tel: 212-376-3200 Fax: 212-376-3280	TICTOC SOFTAUDIT
Izar Associates, Inc.	4 Emery Avenue Randolph, NJ 07869 Tel: 201-442-0577 Fax. 201-442-0633	Fieldex 2000
Mainware, Inc.	7176 Pioneer Creek Road Maple Plain, MN 55359 Tel: 612-932-9154 Fax: 612-932-9155	HourGlass 2000
Micro Focus	2465 East Bayshore Road Palo Alto, CA 94303 Tel: 800-318-4259 Tel: 415-856-4161 Fax: 415-856-6134	Challenge 2000
Namtrig Incorporated	P.O. Box 160757 Altamonte Springs, Florida 32716 Tel. 407-884-6771 Fax: 407-884-6755 e-mail: namtrig@iag.net	ADAMS/400
PRINCE Software, Inc.	1000C Lake Street Ramsey, NJ 07446 Tel. 800-934-2022 FAX: 201-934-0220 e-mail: PORTAL2000@ PRINCESOFTWARE.com	PORTAL 2000 / SIMULATE 2000 PORTAL 2000 / SURVEY 2000 PORTAL 2000 / TRANSLATE 2000
Princeton Softech	1060 State Road Princeton, NJ 08540 Tel: 609-497-0205 FAX: 609-497-0302 e-mail: INFO@PrincetonSoftech.com	Version Merger Relational Tools Suite

Figure 7-5 (Page 3 of 4). Solution Developer Tools

Company Name	Contact	Product Name
Quintic Systems, Inc.	3166 Des Plaines Avenue, Suite 36 Des Plaines, Illinois 60018 Tel. 800-699-1169 Tel 708-699-1169 Fax. 708-699-1214	Century File Conversion Century Source Conversion
Reasoning Systems	3260 Hillview Avenue Palo Alto, CA 94304 Tel 415-494-6201 Fax 415-494-8053	Software Refinery REFINE/COBOL
SEEC, Inc.	5001 Baum Blvd Pittsburgh, PA 15213 Tel: 412-682-4991 Fax 412-682-4958	COBOL Analyst
Software Eclectics, Inc.	Suite 131 10955 Jones Bridge Rd Alpharetta, GA 30202-7343 Tel 800-457-3113 Tel 770-667-9117 Fax 770-667-9417	SE/One
Software Migrations Limited	Unit 1C Mountjoy Research Center Stockton Road Durham, DH1 3SW United Kingdom Tel. 44+(0)191 386 0420 Fax 44+(0)191 386 1243	FermaT 2000
The Source Recovery Co., Inc.	992 E. Freeway Drive, Suite A Conyers, GA 30207-5916 Tel 770-785-9801 FAX 770-760-7316 e-mail: 72604,2340@Compuserve com	Source Recovery Services
Tech-Beamers	21 Davis Ave Poughkeepsie, NY 12603 Tel 914-473-4444 FAX 914-473-4478	Year 2000 S/390 Workstation
TechForce BV	Saturnusstraat 60 P O Box 3108 2132 HB Hoofddorp The Netherlands Tel: +31.23 56.22929 Fax: +31.23.56 27052 e-mail: sales@techforce nl	COSMOS Year 2000 Workbench
TransCentury Data Systems	111 Pine Street, Suite 715 San Francisco, CA 94111 Tel 415-255-7082 Fax 415-255-4584	TransCentury Calendar Routines
VIASOFT	3033 N 44th Street Phoenix, AZ 85018 Tel. 800-525-7775 Tel 602-952-0050 Fax 602-840-4068	Enterprise 2000

Figure 7-5 (Page 4 of 4) Solution Developer Tools

Company Name	Contact	Product Name
Visionet Systems Inc.	P O Box 316 1103 Pleasant Rd Monmouth Jct , NJ 08852 Tel 908-329-8090 Fax 908-329-1712 e-mail visionet@superlink.net	Millennium/400

Chapter 8. IBM Consulting and Services

The following is extracted directly from IBM announcement letter 25744 dated 31 October 1995. See that IBM document for complete announcement information

TRANSFORMATION 2000: IBM's Century Date Change Solutions

IBM recognizes that the Year 2000 change poses a significant challenge for the Information Technology industry To help with this change IBM's Information Systems Solution Corporation (ISSC) has developed a comprehensive set of solutions that takes into account applications, systems software and hardware in both centralized and distributed environments This comprehensive solution set is called TRANSFORMATION 2000.

The TRANSFORMATION 2000 approach seeks to balance Year 2000 investment activities with current and planned strategic initiatives (for example, new architectures and new application development) ISSC brings together state-of-the-art techniques and technologies developed and proven through both internal and external projects for the Year 2000 and other data field expansions This experience enables ISSC to help reduce both the cost and the complexity of implementing the Year 2000 change.

TRANSFORMATION 2000 Solutions

Assessment and Strategy

The Assessment and Strategy solution is an eight- to twelve-week engagement that will identify the magnitude of the Year 2000 problem. This solution covers a comprehensive range (5 to 150+ million lines of code) of portfolios in a mixed language environment ISSC's approach allows for analysis across the entire client portfolio thereby identifying logical work partitions from which the implementation plan will be formulated, helping to reduce the overall risk of the project.

ISSC uses a optimal combination of commercially available and internally developed tools depending on the client environment.

At the completion of the project, the client will have a documented strategy, general cost estimates, timeframes, and resource estimates required to implement the Year 2000 change

Detailed Analysis and Planning

The Detailed Analysis and Planning solution provides an in-depth analysis of each of the Year 2000-affected areas and identifies detailed work partitions needed to maximize the potential for automated change This review includes not only the application code, but file structures and other environmental objects

The output from this engagement includes detailed work partitions, detailed work and resource plans, and the customized specifications needed to implement the Year 2000 changes

© Copyright IBM Corp 1995, 1996

Implementation

The Implementation solution is unique and consists of automating the required changes to source code and data, testing, and production turnover. In addition, TRANSFORMATION 2000 includes an optional conditioning activity which will improve the overall quality of the resultant application as well as reduce the total time required for the Year 2000 changes. Conditioning includes but is not limited to the following

- standardization/rationalization of data definitions
- segmenting procedure logic
- encapsulating date handling/developing common routines
- externalizing hard-coded data
- restructuring COBOL procedure logic
- removing system redundancy

Also included in the TRANSFORMATION 2000 solution are sophisticated testing activities

1. After making the Year 2000 changes, the total environment must be tested to ensure it can handle 19xx data correctly.
2. While running in 199x, the system must be able to handle dates after 1999.
3. When running in the Year 2000 and after, the system needs to be tested to ensure it can handle both 19xx and 2xxx data correctly

Year 2000 Clean Management

During the Year 2000 change process it is critical for the client to protect the investment of the application modifications. When enhancements are incorporated into the application and/or new releases of software products are implemented, the Year 2000-clean status must be maintained. For this reason, ISSC has developed the Year 2000-Clean Management solution which focuses on instituting the appropriate processes, disciplines, techniques, and technologies in order to preserve the Year 2000 investment.

Summary

Recognizing that the Year 2000 is a significant client challenge, ISSC has created a Year 2000 Center-of-Competence (CoC) function, with the initial site in Atlanta. The CoC supports ISSC and IBM client services teams in providing complete Year 2000 solutions to our clients

Further information on the TRANSFORMATION 2000 solution set is available from your IBM or ISSC representative.

Appendix A. IBM Year2000-Ready Key Program Products and Hardware

This section presents key IBM products and offerings and their level or levels that are **currently available or planned to be available** as Year2000 ready by year-end 1996

These lists are not meant to be exhaustive or exclusive, but are provided to answer the most common questions This list will be updated periodically as more information becomes available

Specific program products are listed under the IBM platform they support as follows.

- MVS
- TPF
- VSE/ESA
- VM
- OS/400
- AIX
- OS/2
- Lotus Products

If a product is listed with a version/release number, it indicates that level of the product (with PTF if applicable) and all subsequent versions/releases will include Year2000 support Keep in mind that new versions will have new product numbers If just a version is listed or if no version/release number is listed, you can assume that the most recent version/release as of year-end 1996 will include the Year2000 support Be aware that if the most current level at the end of 1996 is a different version from the one listed here, it will have a different product number

Exception Considerations and Program Products

There will be a few exceptions to the general rule, typically where a product will be superseded or replaced in the relevant time period, and incorporation in the lists does not imply that a product or version of a product will necessarily be current or available in 1999 or beyond Where a product is superseded or replaced by a new version/release by the end of 1996, the later level might not be listed here, rather only the older version When it is appropriate to publish the later level, the list will indicate the level it has replaced

Environmental Report Editing and Printing (EREP)

When using Environmental Report Editing and Printing (EREP) (5654-260, EREP for VM, 5656-260, EREP for VSE. and 5658-260, EREP for MVS), to produce reports where the records have been created in both 1999 and 2000, special handling is necessary to prevent problems and minimize customer impact. Because EREP data on a single LOGREC tape spans a relatively short period of time, be aware of the following limitations when processing EREP reports which include both 31 December 1999 and 1 January 2000

- Input of the DATE parameter is restricted to a 2-digit-year specification. If you do not use the DATE parameter, and the data on a given LOGREC tape

© Copyright IBM Corp 1995, 1996

includes from both 31 December 1999 and 1 January 2000, the output is sorted with data from the year 1999 after the data from the year 2000.

- EREP does not properly process date selection if the range includes both 31 December 1999 and 1 January 2000 Therefore, if you desire to select report entries for such a range, you must create two reports. On the first report specify `DATE=(yy.ddd,99.365)` and on the second report specify `DATE=(00.001,yy.ddd)`

Your IBM representative can also provide detail on the Year2000 enhancements made to many of these products and the potential impact to your operating environment if you choose to use a prior version or release that is not Year2000 ready

MVS

Figure A-1 (Page 1 of 5) MVS Platform Products

Program Product Number	**Product Name (Version/Release)**
5668-854	ACF/NCP V4 R3 M1
5665-338	ACF/SSP V3 R6
5695-117	ACF/VTAM 4 3
5688-216	AD/Cycle C/370 V1R2
5688-194	AD/Cycle CODE/370 V1R2
5648-020	ADSTAR Distributed Storage Manager/MVS V1 R1
5655-119	ADSTAR Distributed Storage Manager/MVS V2 R1
5688-021	AFP Bar Code V1 R1
5648-113	AFP Font Collection V1 R1
5695-068	Airline Control System (ALCS) V2 R1
5685-151	AOC/MVS V1 R2
5688-228 5688-229	APL2 V2 R2
5665-348	Application Development Facility II V2 R2
5655-002	Application Support Facility V3 R1
5648-018	Application System V3 R2
5655-065	Batchpipes/MVS V1 R2
5695-045	BookManager Build V1 R3
5695-046	BookManager Read V1 R3
5688-224	Browsemaster V2
5665-279	BTAM V1 R1
5665-264	Bulk Data Transfer (BDT) V2 R1
5635-001	CADAM Interactive Design System (and associated products)
5627-COM	CATIA Object Manager V4 1.6 (and associated products)
5734-F11	Check Processing Control System (CPCS) V1 R11 M0
5655-018	CICS for MVS/ESA V4 1
5695-061	CICS Application Migration Aid V1 R1
5665-463	CICS/Distributed Data Management(DDM) V1 R1 M0
5696-582	CICS Transaction Affinities Utility V1 R1
5695-081	CICSPlex System Mgr V1.2
5695-010	CICSVR V2
5688-197	COBOL for MVS & VM V1.2
5655-121	C/C++ for MVS/ESA V3 R2
5688-187	C/370 Compiler V2R1
5688-188	C/370 Library V2R2
5798-DLO	Data Base Edit Facility (DBEDIT) V1 R3
5787-LAR	Data Base Integrity Control Facility (DBICF) V4 R1

MVS

Figure A-1 (Page 2 of 5) MVS Platform Products

Program Product Number	Product Name (Version/Release)
5685-036	DataInterchange/MVS V1 R5
5695-076	DataInterchange/MVS CICS V1 R5
5655-076	Data Propogator (DPROPR)/CAPTURE V1 R2 M0
5622-267	Data Propogator (DPROPR)/APPLY V1 R2 M0
5685-DB2	DB2 V3 R1
5688-515	DB2 Administration Tool V1 R1
5695-077	DB2 Automated Utility Generator V1 R2
5655-102	DB2 Performance Monitor(PM) V4 R1
5740-XC5	Development Mgmt System/CICS V1 R5
5655-257	Device Support Facility (ICKDSF) V1 R16
5747-DS1	Device Support Facility (ICKDSF) Standalone V1 R16
5695-DF1	DFSMS/MVS V1 R2 (DFSMSdfp, DFSMSdss, DFSMShsm, DFSMSrmm, NFS feature, DFSMSopt feature)
5740-SM1	DFSORT V1 R13
5655-068 5655-069	DFS/MVS V1 1
5665-290	DISOSS V3 R4
5685-101	Displaywrite/370 V2 R1 under MVS/TSO
5685-101	Displaywrite/370 V2 R1 under MVS/CICS
5648-106	Distributed Security Manager for MVS V1 R1
5655-103	DITTO/ESA for MVS V1 R1
5748-XX9	Document Composition Facility (DCF) V1 R4
5748-XXE	Document Library Facility (DLF) V1 R3
5688-168	dpAccounting Mgr (DPAM) for MVS V1.2
5735-XXB	Emulation Program V1 R6 M1
5695-101	Enterprise Performance Data Manager (EPDM) V1 R1
5688-008	ES Connection Manager V1 R3
5655-073	FASTService for MVS Runtime Library V1 R2
5685-088	FASTService for MVS V1 R2
5655-123	FFST Customized Services V1 R2
5655-129	Flowmark for MVS V1 R1
5668-890	Font Library Service Facility (FLSF) V1 R1
5668-806	FORTRAN V2.5 Compiler/Library/Interactive Debug
5688-087	FORTRAN V2.6 Compiler/Library
5668-805	FORTRAN V2 6 Library only
5668-723	GDDM/IVU V1 R1
5668-801	GDDM/IMD V2 R1
5668-812	GDDM/PGF V2 R1
5688-113	GDDM/OS2 Link V1 R0

Figure A-1 (Page 3 of 5). MVS Platform Products

Program Product Number	Product Name (Version/Release)
5665-356	GDDM/MVS V2 3
5695-167	GDDM/MVS V3 1.1
5740-XX7	Generalized Information System (GIS) V1 R1 M1
5685-105	geoManager V1 R1 M5
5764-023	geoManager API V1 R1 M0
5799-AXX	Graphics Attachment Support Program(GASP) V1 R3 M10
5688-169	Graphical Display and Query Facility V2 R2 M1
5688-050	Graphics Program Generator V2
5697-119	Hardware Configuration Manager (HCM) V1 R1
5655-068 5655-069	Hardware Configuration Definition (HCD) V5 R1
5696-234	HLASM for MVS & VM & VSE V1 R2
5685-037	Host Support Product (HSP)
5648-001	IGES Processor V3 R1 M2
5695-041	ImagePlus for MVS/Folder Application Facility V2 R2
5695-042	ImagePlus for MVS/Object Distribution Manager V2 R2
5655-071	ImagePlus VisualInfo Library Server for MVS/ESA
5655-072	ImagePlus VisualInfo Object Server for MVS/ESA
5695-176	IMS/ESA 5 1
5798-CHJ	IMSASAP V1 R1
5655-085	IMS Compress V1 R2
5685-093	IMS Data Base Tools (DBT) V2 R1
5655-109	IMS Data Entry Databases (DEDB) Fast Recovery (DFR) V1 R1
5740-XXR	IMS DB Analyser V1 R1
5655-038	IMS Message Requeuer
5798-COP	IMS Performance Analysis and Reporting System (IMSPARS) V1 R1
5668-948	IMS/VS Batch Terminal Simulator (BTS) V2 R2
5695-171	Info Management V6 R2
5665-393	Inforem Base
5665-394	Inforem Allocation
5685-051	Integrated Cryptographic Service Facility (ICSF)/MVS V1.2
5685-054	ISPF V3 R1
5695-080	Item Access Fac (IAFC) V1 R3
5785-BAC	JES/328X Print Facility V2 R2
5655-068	LAN Server for MVS
5688-198	Language Environment for MVS & VM V1 R4
5695-123	LANRES/MVS
5655-039	MERVA/ESA V3 R2 (supports the definitions in the SWIFT User Handbook)
5655-040	MERVA/MVS V3 R2 (supports the definitions in the SWIFT User Handbook)

MVS

Figure A-1 (Page 4 of 5) MVS Platform Products

Program Product Number	**Product Name (Version/Release)**
5695-137	MQSeries for MVS/ESA V1.1.3
5655-068	MVS/ESA SP JES2 V5 R1
5655-069	MVS/ESA SP JES3 V5 R1 M1
5655-ACS	NaviQuest V1 R1
5665-365	NetView Access Services V1.3.2
5695-036	NetView Access Services V2 for MVS
5695-169	NetView AutoBridge
5685-016	NetView Distribution Manager (DM) V1.6
5685-108	NetView FTP for MVS V2.2
5655-044	NetView MultiSystem Manager V1 R1
5655-126	NetView MultiSystem Manager V2 R1
5655-007	NetView V3 R1
5685-111	NetView V2.4
5665-333	NetView Performance Monitor V1.6
5655-043	NetView Performance Monitor V2.1, 2.2
5696-583	NetView Remote Ops Manager
5685-045	Network Design & Analysis(NetDA) V1 R2
5735-XX7	Network Terminal Option (NTO) V1 R5 M1
5648-077	NtuneMON V1 R1
5648-089	NTuneNCP V1 R1
5685-106	OfficeVision/MVS 1.3
5695-007	Operations Planning and Control (OPC)/ESA V1.3.1
5655-064	OpenEdition DCE Appl Support for MVS/ESA V1.1
5668-909 5668-910 5668-911	OS/VS PL/I V2 R3
5740-RG1	OS/VS RPGII V1 R1
5645-001	OS/390 V1 R2
5655-104	Open Systems Adapter Support Facility V1 R1
5688-191	Overlay Generation Language (OGL/370) V1.1.0
5688-190	Page Printer Formatting Aid (PPFA) V1 R1
5668-767	PASCAL
5665-311	PC/File Transfer V1 R1 M2
8641-MYF 8641-MYC 8641-MYS 8641-NYV	PC Server 500 System/390
5695-101	Performance Reporter for MVS V1.2
5771-ABC	Pi and Specials V1 R1
5688-235	PL/I for MVS & VM V1 R1

Figure A-1 (Page 5 of 5) MVS Platform Products

Program Product Number	**Product Name (Version/Release)**
5665-307	Print Management Facility (PMF) V1 R1
5665-340	Print Services Access Facility (PSAF) V1 R1
5655-010	ProductManager Application Svcs Mgr (and associated products) V3 1
5706-236	PROLOG
5695-040	PSF/MVS V2 R2
5688-015	Publishing Systems BookMaster V1 R4
5668-721	QMF/MVS V2 R4
5695-039	RACF V2 R2
5655-067	ReDiscovery/MVS V1 R1
5648-048	Report Management and Distribution System (RMDS) V2 R2
5695-013	REXX Compiler for 370 V1 R3 M0
5695-014	REXX Runtime Library V1 R3 M0
5655-086	REXX Development System for CICS/ESA V1 R1 M0
5655-087	REXX Runtime Facility for CICS/ESA V1 R1 M0
5655-084	RMF V5 R1
5665-366	Screen Definition Facility (SDF) II MVS V1 R4
5665-488	SDSF V1 R6
5695-070	SearchManager/370 under CICS/MVS V1 R2
5665-397	Service Level Reporter V3 3 1
5668-949	SMP/E 1 8 0
5668-949	SMP/E 1 8 1
5665-345	SNA Application Monitor (SAMON) V1 R0
5696-822	SOMobjects for MVS V1 R1
5655-146	SystemView for MVS Base V1 R1
5695-178	SystemView Automated Operations Network (AON) V1.1
5735-RC3	TCAM V2 R4
5665-314	TCAM V3 R1
5655-HAL	TCP/IP
5798-DYE	The Information Facility (TIF) 2 1.3
5688-121	TPNS V3 R4
5752-VS2	TIOC V1 R1
5688-139	Target System Control Facility(TSCF) V1 R2
5685-025	TSO/E V2 R4
5734-UT1	TSO Data Utilities
5648-078	VisualGen Host Services V1 R1 (referred to previously as CSP)
5648-109	VisualLift for MVS, VSE & VM V1 R1 M1
5748-AP1	VS APL V1 R4
5748-XX1	VS BASIC
5668-719	X 25 NPSI V2 R1

TPF

Figure A-2. TPF Platform Products

Program Product Number	Program Name (Version Release)
5748-T14	TPF 4.1
5695-068	TPF/MVS 2 1.3
5706-196	TPF/DF 1.1.3
5695-067	EOCF/2 1.0 with CSD2

VSE/ESA

Figure A-3 (Page 1 of 2) VSE Platform Products

Program Product Number	Program Name (Version/Release)
5750-ACD	VSE/ESA (Package) 1.4
5690-VSE	VSE/ESA (Package) 2.1
5686-032	VSE/Advanced Functions 5.2
5686-066	VSE Central Functions 6.1
5686-034	VSE/FAST COPY 2.2
5686-036	VSE/ICCF 3.2
5656-092	VSE/OLTEP 1.1
5686-033	VSE/POWER 5.2
5686-028	VSE/SP Unique Code 5.2
5686-037	VSE/VSAM 2.2
5666-363	ACF/VTAM for VSE/ESA 3.4
5686-065	ACF/VTAM for VSE/ESA 4.2
5746-RC5	BTAM/ES 1.1
5686-026	CICS for VSE/ESA V2.3
5688-052	DITTO for VSE and VM 3.2
5648-099	DITTO/ESA for VSE V1 R1
5746-XC4	Development Management System(DMS)/CICS/VS V1 R5 M0
5686-073	ADSTAR Distributed Storage Manager for VSE
5686-072	ALERT for VSE 4.9
5686-079	ALERT for CICS/VSE 4.9
5688-187	C/370 Compiler V2R1
5688-188	C/370 Library V2R1
5785-CCC	CCCA/VSE
5798-DYE	COBOL Report Writer Precompiler
5686-068	IBM COBOL for VSE/ESA (COBOL/VS) 1.1
5746-SM3	DFSORT/VSE V3 R2
5746-XX1	DL/I DOS/VS 1.10
5686-041	VSE/DSNX 2.1
5686-057	GDDM/VSE V3.1.1
5666-328	GDDM/VSE V2.3
5696-234	HLASM for MVS & VM & VSE V1 R2
5747-DS2	Device Support Facility (ICKDSF) R16
5747-DS1	ICKDSF Stand-alone
5686-067	Language Environment for VSE/ESA 1.1
5686-063	MERVA V3 R2 (supports the definitions in the SWIFT User Handbook)
5686-055	NetView for VSE/ESA V2.3
5746-XC5	VSE/OCCF 1.4

VSE/ESA

Figure A-3 (Page 2 of 2) VSE Platform Products

Program Product Number	Program Name (Version/Release)
5686-069	IBM PL/I for VSE/ESA 1 1
5686-040	PSF/VSE 2 2
5648-061	QMF/VSE V3 R2
5686-058	REXX/VSE 1 1
5746-RG1	DOS/VS RPG II 1 3
5746-XXT	Screen Definition Facility (SDF)/CICS V1 R5
5688-103	SQL/DS 3.5
5648-078	VisualGen Host Services V1 R1
5648-086	VisualGen Generator Option for VSE (replaced by VisualGen V2 R0)
5648-109	VisualLift for MVS, VSE & VM V1 R1 M1

VM

Figure A-4 (Page 1 of 2). VM Platform Products

Program Product Number	**Product Name (Version/Release)**
5648-063	ACF/NCP 7.3 0
5654-010	ACF/VTAM 4 2.0
5688-216	AD/Cycle C/370 1.2 0
5688-194	AD/Cycle CODE/370 V1R2
5648-020	ADSTAR Distributed Storage Manager/VM
5654-033	C for VM/ESA Compiler V3R1
5635-001	CADAM Interactive Design System (and associated products)
5627-COM	CATIA Object Manager (and associated products)
5688-197	COBOL for MVS & VM
5688-187	C/370 Compiler V2R1
5688-188	C/370 Library V2R2
5684-042	Device Support Facility (ICKDSF) R16
5747-DS1	ICKDSF Stand-alone
5684-112	DFSMS/VM 2.2 1
5684-134	DFSORT/CMS V2 R1
5748-XE4	DIRMAINT 1 5.0
5684-113	Display Mgt System for CMS 2 1.0
5654-029	DITTO/ESA for VM V1 R1
5668-806	Fortran 2 6 0
5684-168	GDDM/VM 3 1 1
5684-007	GDDM/VM XA V2 3
5696-234	HLASM for MVS & VM & VSE V1 R2
5684-157	Host Mgt Facilities 1.1 1
5684-043	ISPF 3 2 0
5648-039	LAN File Services (LFS) 1.1 2
5688-198	Language Environment for MVS & VM
5684-142	LANRES/VM 1 3.0
5684-028	NetView Access Services VM V1.3.2
5684-001	NetView Performance Monitor/VM V1 6
5684-011	NetView Performance Monitor/VM V2
5756-051	NetView for VM/ESA 2.3
5684-084	OV/VM 1 3 0
5668-909	OS PL/I Version 2
5688-235	PL/I for MVS and VM
5664-310	PMF/VM Version 1.1
5664-312	PSAF/VM Version 1 1
5684-141	PSF/VM Version 2.1.1

Figure A-4 (Page 2 of 2) VM Platform Products

Program Product Number	**Product Name (Version/Release)**
5684-100	PVM 2.1 1
5706-255	QMF V3 R1 M1
5740-XXH	RACF
5695-013	REXX Compiler V1 R3 M0
5684-096	RSCS 3 2 0
5664-307	Screen Definition Facility (SDF) II VM V1 R3
5684-143	SearchManager/370 for VM V1 R2
5688-103	SQL/DS 3 5
5735-FAL	TCP/IP 2.3 0
5688-121	TPNS
5648-109	VisualLift for MVS, VSE & VM V1 R1 M1
5654-030	VM/ESA Version 2
5686-037	VSE/VSAM for VM 6 1.0

OS/400

Figure A-5 (Page 1 of 3) OS/400 Platform Products

Program Product Number	Product Name (Version/Release)
5716-SV1	ADSTAR Distributed Storage Manager for OS/400 V3 R6
5763-SV1	ADSTAR Distributed Storage Manager for OS/400 V3 R1
5733-197	ADSTAR Distributed Storage Manager for OS/400 V2 R3
5716-AP1	Adv DBCS Printer Support for OS/400 V3
5763-AP1	Adv DBCS Printer Support/400 V3
5716-AF1	Adv Function Printing Utils for OS/400 V3
5716-FNT	Adv Function Printing Fonts for OS/400 V3
5716-FN1	Adv Function Printing DBCS Fonts for OS/400 V3
5763-AF1	Adv Function Printing Utils/400 V3
5763-FNT	Adv Function Printing Fonts/400 V3
5763-FN1	Adv Function Printing DBCS Fonts/400 V3
5716-PW1	Application Development ToolSet for OS/400 V3
5763-PW1	Application Development ToolSet/400 V3
5716-CL1	Appl Dev ToolSet Client Svr for OS/400 V3
5763-CL1	Appl Dev ToolSet Client Svr/400 V3
5716-PD1	Application Program Driver for OS/400 V3
5763-PD1	Application Program Driver/400 V3
5763-DS1	AS/400 Business Graphics Util V3
5763-DB1	AS/400 System/38 Utilities V3
5763-VR1	AS/400 VRPG Client/2 V3
5716-BR1	Backup Recovery & Media Svcs for OS/400 V3
5763-BR1	Backup Recovery and Media Services/400 V3
5716-DS1	Business Graphics Utility for OS/400 V3
5716-CP2	CallPath for OS/400 V3
5763-CP2	CallPath/400 V3
5716-DFH	CICS for OS/400 V3
5716-XA1	Client Access for OS/400 Version 3
5716-XC1	Client Access Windows+ Client for OS/400 Version 3
5716-XL1	Client Access DOS Client for OS/400 Version 3
5716-XM1	Client Access ToolKit for OS/400 Version 3
5716-XB1	Client Access DOS with ExtMem Client for OS/400 V3
5716-XF1	Client Access OS/2 Client for OS/400 V3
5716-XG1	Client Access Opt OS/2 Client for OS/400 V3
5716-US1	Client Access Ultimedia Tools for OS/400 V3
5763-US1	Client Access/400 Ultimedia Tools V3
5763-XA1	Client Access/400 Family Version 3
5763-XC1	Client Access/400 for Windows 3.1 Version 3

Figure A-5 (Page 2 of 3) OS/400 Platform Products

Program Product Number	Product Name (Version/Release)
5763-XL1	Client Access/400 for DOS Version 3
5763-XM1	Client Access/400 ToolKit Version 3
5763-XB1	Client Access/400 for DOS with Ext Mem V3
5763-XF1	Client Access/400 for OS/2 Client V3
5763-XG1	Client Access/400 Optimized for OS/2 V3
5716-CM1	Communications Utils for OS/400 V3
5763-CM1	Communications Utils/400 V3
5763-CD1	CoOperative Development Environment/400 V3
5716-CR1	Cryptographic Support for OS/400 V3
5763-CR1	Cryptographic Support/400 V3
5716-DP1	DataPropagator Rel Capture&Apply for OS/400 V3
5763-DP1	DataPropagator Rel Capture&Apply/400 V3
5716-ST1	DB2 Query Manager and SQL Development Kit for OS/400
5763-ST1	DB2/400 Query Manager and SQL Development Kit Version 3
5798-TAY	Facsimile Support for OS/400
5798-RZT	Facsimile Support/400
5716-CX2	Integrated Language Environment C for OS/400 Version 3
5763-CX2	Integrated Language Environment C/400 Version 3
5716-CB1	Integrated Language Env COBOL for OS/400 V3
5716-RG1	Integrated Language Env RPG for OS/400 V3
5763-CB1	Integrated Language Environment COBOL/400 V3
5763-RG1	Integrated Language Env RPG/400 V3
5716-JS1	Job Scheduler for OS/400 Version 3
5716-DCT	Language Dictionaries for OS/400 V3
5763-DCT	Language Dictionaries/400 V3
5763-MW1	ManageWare/400 V3
5763-MQ1	MQSeries for OS/400 V3 1
5798-TBC	Mobile Network Access PagerPac for OS/400
5798-TBD	Mobile Network Access RadioPac for OS/400
5733-196	NetView FTP V3 1 for OS/400
5798-TBA	Neural Network Util for OS/400
5798-RZK	Neural Network Util/400
5798-TAQ	OfficeVision JustMail for OS/400
5798-RZJ	OfficeVision JustMail/400
5716-WP1	OfficeVision for OS/400 V3
5763-WP1	OfficeVision/400 V3
5716-SS1	Operating System/400 V3
5763-SS1	Operating System/400 V3
5716-OS1	OSI Comm Subsystem for OS/400 V3

Figure A-5 (Page 3 of 3) OS/400 Platform Products

Program Product Number	Product Name (Version/Release)
5763-OS1	OSI Comm Subsystem/400 V3
5716-FS1	OSI File Svcs for OS/400 V3
5763-FS1	OSI File Svcs/400 V3
5716-MS1	OSI Message Svcs for OS/400 V3
5763-MS1	OSI Message Svcs/400 V3
5798-RYZ	PagerPac for the AS/400
5716-PM1	Performance Management for OS/400 V3
5716-PT1	Performance Tools for OS/400 V3
5763-PT1	Performance Tools/400 V3
5716-CF1	Point-Of-Sale Comm Util for OS/400 V3
5763-CF1	Point-Of-Sale Comm Util/400 V3
5716-QU1	Query for OS/400 Version 3
5763-QU1	Query/400 Version 3
5798-RYY	RadioPac for the AS/400
5733-218	Report/Data Archive&Retrieval Sys for OS/400
5716-STK	SOMobjects Developer Toolkit for OS/400 V3
5716-DB1	System/38 Utilities for OS/400 V3
5716-MG1	SystemView Managed Sys Serv for OS/400 V3
5763-MG1	SystemView Managed System Services/400 V3
5763-MW1	SystemView ManageWare/400 V3
5716-ES1	SystemView OMEGAMON Svcs for OS/400 V3
5763-ES1	SystemView OMEGAMON Svcs/400 V3
5716-SM1	SystemView System Manager for OS/400 V3
5763-SM1	SystemView System Manager/400 V3
5716-TC1	TCP/IP Connectivity Utils for OS/400 V3
5763-TC1	TCP/IP Connectivity Utils/400 V3
5716-UB1	Ultimedia Business Conf'g for OS/400 V3
5763-UB1	Ultimedia Business Conf'g/400 V3
5716-VG1	VisualGen Host Svcs for OS/400 V3
5763-VG1	VisualGen Host Svcs/400 V3
5716-CX4	VisualAge C++ for OS/400 V3

AIX

Figure A-6 (Page 1 of 4) AIX Platform Products

Program Product Number	Product Name (Version/Release)
5765-564	ADSTAR Distributed Storage Manager V2 R1
5697-078	ADSTAR Distributed Storage Manager V1 R2.1
5765-268	AIX Asynch.Term.Svr.Accelerator/6000
5765-266	AIX CallPath Server/6000
5765-117	AIX DCE Base Services/6000 V1
5765-119	AIX DCE Cell Dir Server/6000 V1
5765-121	AIX DCE Enhanced Dist.File Sys./6000 V1
5765-120	AIX DCE Global Dir.Server/6000 V1
5765-259	AIX DCE Global Dir.Client/6000 V1
5765-118	AIX DCE Security Server/6000 V1
5765-232	AIX DCE Threads/6000 V1
5765-001	AIX DirectTalk/6000
5696-902	AIX Distributed SMIT V2.2 for AIX
5765-042	AIX EngSci Subroutine Lib./6000 V2 R2
5696-708	AIX File Storage Facility (FSF)
5756-030	AIX for RISC System/6000 Version 3.2.5
5696-923	AIX HACMP/6000 V3.1
5696-933	AIX HACMP/6000 V4.1 for AIX V4
5765-551	AIX HIPPI/6000 V3 R2 M3
5696-108	AIX InfoCrafter/6000 V1.1
5696-893	AIX InfoCrafter V2.1
5621-107	AIX NetView Service Point 1.2
5621-013	AIX OSL/6000 Version 1.2
5765-296	AIX Parallel System Support Programs V1.2
5765-529	AIX Parallel System Support Programs for AIX V2.1
5696-899	AIX Performance Aide/6000 V2.1
5696-900	AIX Performance Toolbox/6000 V2.1
5765-349	AIX /SMARTsort for Workstation
5765-261	AIX SNA Gateway/6000 V2.2
5696-906	AIX Ultimedia Services/6000 V2.1.1
5765-393	AIX Version 4.1.3
5696-868	AIX X.25 V1.1
5765-011	AIX X-Windows 3270 Emulator/6000
5601-248	AIX XL FORTRAN/6000 Version 2.3
5765-018	AIX XL Fortran Compiler/6000
5765-019	AIX XL Fortran Runtime Env/6000
5765-176	AIX XL Fortran for AIX V3.2

Figure A-6 (Page 2 of 4). AIX Platform Products

Program Product Number	**Product Name (Version/Release)**
5601-251	AIX XL Pascal Runtime Env./6000 (service withdrawn April 1995)
5601-254	AIX XL Pascal Compiler/6000 Version 1.1.2 (service withdrawn April 1995)
5765-245	AIX XL Pascal Compiler/6000 Version 2 (replaces 5601-251 and 5601-254 above)
5601-260	AIX 3270 Host Conn/6000 Version 1.3
5765-249	AIX 5080 Emulation Program/6000
5765-398	AIXlink V2.1.1
5696-926	AIXlink/X.25 V1 1
5696-904	AIXwindows Display Postscript V1 1
5601-257	AIXwindows Environment/6000 V1 2.5
5756-027	AIXwindows Interface Composer/6000 (AIC) V1.2
5765-423	C for AIX
5765-421	C Set + + for AIX
5626-COM	CATIA Object Manager V4 R1 M6 (and associated products)
5765-148	CICS for AIX V1 R2
5765-152	CICS Client for AIX V1 R2
5765-427	CICS System Manager V1 1 for AIX
5801-AAR	COBOL Set V1.1.0 for AIX
5765-561	CommonPoint Application System for AIX V1.1
5765-562	CommonPoint Application Development Toolkit V1 1
5765-652	Communications Server for AIX V4
5765-022	Consumer Transaction Definition/6000
5767-023	Consumer Transaction Runtime/6000
5765-418	Data Encryption Std Lib Routine V1.1
5765-642	Database Server V4
5765-256	DataHub Support/6000 V3.2 (service withdrawn October 1995)
5801-AAR	DataHub for UNIX Operating Systems (replaces 5765-256 above)
5871-AAA	DB2 for AIX V2.1
5765-459	DB2 SDK/6000 V1 R2
5765-453	DB2 SDK for AIX V2 R1
5765-464	DB2/6000 V1 R2
5871-AAA	DDCS for AIX V2.3
5696-239	Encina Monitor for AIX/6000
5696-238	Encina PTP Exec for AIX/6000
5696-347	Encina PTP Gateway for AIX/6000
5696-240	Encina Server for AIX/6000
5696-237	Encina Structured File Svr for AIX/6000
5765-527	Extended Systems Administration Feature (SystemView)

Figure A-6 (Page 3 of 4) AIX Platform Products

Program Product Number	**Product Name (Version/Release)**
5696-923	High Availability Cluster Multi-Processing V3 R1 M1
5765-482	LANDP/6000 V2 1
5801-AAR	FlowMark for AIX V2 R2 (replaces 5765-270 FlowMark for AIX V1 R1)
5765-026	geoGPG/6000
5696-919	Hypertext Information Base Libraries V1.1
5696-898	InfoExplorer License Extension V1.1
5765-223	InterMix for AIX Version 1.2
33H4191	Internet Connection Secure Server for AIX
33H4190	Internet Connection Server for AIX
33H4290	Internet Connection Secured Network Gateway V2 1
5765-527	Job Scheduler Feature (SystemView)
5765-264	LAN Management Utilities/6000 V1 1.3
5765-145	LoadLeveler Version 1 2 1
5765-449	MERVA AIX V1 R1 (supports the definitions in the SWIFT User Handbook)
5765-115	MQSeries for AIX Version 2 1.0 (service to be withdrawn 31 Dec 1996 Announced 28 Mar 1995, letter number 295-144)
5765-550	NetBIOS & IPX/SPX Support/6000 V2
5871-BBB	NETSP Secured Network Gateway 1 2 (feature 6362)
5765-196	NetView Distribution Manager for AIX V1.1 (service to be withdrawn Sep 1996 Announced 5 Dec 1995, letter number 995-140)
5765-214	NetView Distribution Management Agent for AIX V1.1 (service to be withdrawn Sep 1996 Announced 5 Dec 1995, letter number 995-140)
5696-731	NetView for AIX 3.1
5622-242	NetView FTP Client V1 1 for AIX
5765-435	NetView FTP Server V1 1 for AIX
5696-236	Netware for AIX/6000
5696-939	OpenGL & GL 3 2 Version 4 1 3
5765-352	OpenMail for AIX B.02 (withdrawn from service Jan 1996 Announced 10 Oct 1995, letter number 995-090)
5765-543	Parallel Environment for AIX V2 1
5765-422	Parallel ESSL for AIX V1 1
5765-297	Parallel I/O File System V1.1
5765-392	Parallel OSL (OSLp) V1 1 1
5765-469	Parallel Visual Explorer
5765-527	Performance Reporter Feature (SystemView)
5696-907	Pex & PHIGS Version 4 1 3
5765-193	ProductManager Application Svcs Mgr (and associated products for DB2)
5765-440	ProductManager Application Svcs Mgr (and associated products for Oracle)

Figure A-6 (Page 4 of 4). AIX Platform Products

Program Product Number	**Product Name (Version/Release)**
5765-605	ProductManager Version 3 Release 1 for AIX
5756-094	PROFESSIONAL CADAM Interactive Design (and associated products)
5765-505	PSF for AIX Version 2 1
5765-246	PVMe for AIX V1 R3
5765-544	PVMe for AIX V2.1
5696-038	Realtime Interface Co-Processor AIX Support V1 1 1
5765-444	Recoverable Virtual Shared Disk V1.1
5765-292	RMONitor V1 1 for AIX
5765-343	Router and Bridge Manager/6000 V2.3
5765-233	SNA Manager/6000 V1.1
5765-247	SNA Server/6000 V2.2
5765-410	Systems Monitor for AIX V2 2
5765-527	SystemView for AIX
5697-213	The Multimedia Server for AIX
5765-265	Trouble Ticket V3.2
5765-400	UIM/X V2.8
5696-398	UniTree for AIX/6000 (service withdrawn Dec 1995)
5765-315	WABI V2.0

Notes

- 5706-294, AIX ADA Run-Time Env/6000 Composer and 5706-291, AIX ADA/6000 Version 1.2.3 were withdrawn from service Nov 1994 and have since been transferred to OC Systems, Inc.
- 5756-085, AIX OSI Messaging and Filing/6000 was withdrawn from service Dec 1994 and has been replaced by ObjectStore by Object Design, Inc

OS/2

Figure A-7 (Page 1 of 2) OS/2 Platform Products

Program Product Number	Product Name (Version/Release)
83G8102	OS/2 Warp, Version 3 CD-ROM
83G8100	OS/2 Warp, Version 3 3.5-inch diskette
83G8701	OS/2 Warp with WIN-OS2, Version 3 CD-ROM
83G8700	OS/2 Warp with WIN-OS2, Version 3 3.5-inch diskette
10H9800	OS/2 Warp Connect, Version 3
10H9810	OS/2 Warp Connect with WIN-OS2, Version 3
52G8474	OS/2 LAN Server Entry Version 4 CD-ROM
52G8468	OS/2 LAN Server Entry Version 4 3.5-inch diskette
52G8476	OS/2 LAN Server Advanced Version 4 CD-ROM
52G8475	OS/2 LAN Server Advanced Version 4 3.5-inch diskette
96F8690	IBM DCE SDK for OS/2 and Windows Version 1.0
96F8691	IBM DCE Client for OS/2 Version 1.0
83G9217	IBM PC DOS Version 7 3.5-inch diskette
83G9219	IBM PC DOS Version 7 5.25-inch diskette
89G1342	ADSTAR Distributed Storage Manager/2
87G7776	AnyNet/2 (and associated products)
03H3624	CallPath CallCoordinator/2*
03H3625	CallPath CallCoordinator/2 Server
03H3626	CallPath CallCoordinator/2 Archive
17H78xx	CallPath DirectTalk/2* V2.01
5622-275	CallPath Server/2
5621-159	CallPath SwitchServer/2*
5648-036	CICS for OS/2
79G0257	Communications Manager/2 V1.1 CD-ROM
79G0258	Communications Manager/2 V1.1 3.5-inch diskette
41H2112	DB2 for OS/2 (V2.1 and later) Single-User, CD-ROM
41H2113	DB2 for OS/2 (V2.1 and later) Single-User, 3.5
41H2114	DB2 for OS/2 (V2.1 and later) Server, CD-ROM
41H2115	DB2 for OS/2 (V2.1 and later) Server, 3.5
41H2120	DDCS for OS/2 V2.3 Single-User, 3.5
41H2121	DDCS for OS/2 V2.3 Multi-User Gateway, 3.5
41H2126	DB2 SDK for OS/2 (V2.1 and later), 3.5
29H1040	Distributed Console Access Facility V1.3
33H4189	Internet Connection Server for OS/2 Warp
33H4207	Internet Connection Secure Server for OS/2 Warp
24H3919	LANDP/2 V3
20H1677	License Use Management Application Developer's Toolkit for OS/2

Figure A-7 (Page 2 of 2) OS/2 Platform Products

Program Product Number	**Product Name (Version/Release)**
41H4495	License Use Management Runtime for OS/2
76G7991	LAN NetView Management Utility for OS/2 V1 1
5622-122	MERVA/2 V3 R2 (supports the definitions in the SWIFT User Handbook)
17H7937	MQSeries for OS/2 V2 0
72G6198	Network Door/2 V1
28H3798	NetView Distribution Manager Easy Preparer for OS/2 V1
17H7720	NetView Distribution Manager/2 V2 1 Entry
17H7721	NetView Distribution Manager/2 V2.1 Extended
10H7889	NetView Distribution Manager/2 V2.1 Remote Administrator
79G9845	NetView Distribution Management Agent/2 V1
16H9589	NetView for OS/2 V2.0
16H9610	NetView for OS/2 V2 1
89G1381	NetView File Transfer Program Client/2
89G1384	NetView File Transfer Program Server/2
5622-009	Network Design and Analysis/2 V1
53G3997	Person-to-Person for OS/2
85G8697	Personal Communications/3270 V4.0
85G8661	Personal Communications AS/400 V4 0 for OS/2
85G8805	Personal Communications AS/400 and 3270 V4.0 for OS/2
5622-551	PSF for OS/2 Version 2 00
27H8163	SMARTsort
96F8379	System Performance Monitor/2 Version 2
50H0794	SystemView for OS/2 V1 R1
89H1653	SystemView Server V4 R0
5802-AAR	TCP/IP for OS/2
31H3744	TeamConnection for OS/2
17H7495	VisualAge V2 for OS/2
5622-679	VisualAge C++ for OS/2
31H3678	VisualGen V2 for OS/2
5801-AAR	Workgroup Services for OS/2 and AIX V2

Lotus Products

Figure A-8 (Page 1 of 2) Lotus Products
Program Product Name (Version/Release)
Approach 3 0
Approach 96
cc:Mail client (all platforms) Release 6 0
Freelance Graphics (all versions and platforms)
Lotus Access Unit/cc:Mail for OS/2
Lotus Access Unit/Notes for OS/2
Lotus Directory Synchronization/Banyan
Lotus Directory Synchronization/Callup
Lotus Directory Synchronization/cc:Mail
Lotus Directory Synchronization/Central
Lotus Directory Synchronization/DEC
Lotus Directory Synchronization/EAB
Lotus Directory Synchronization/EMC2
Lotus Directory Synchronization/LMS
Lotus Directory Synchronization/Lotus Messaging Switch
Lotus Directory Synchronization/MEMO
Lotus Directory Synchronization/Microsoft Mail for PC Networks
Lotus Directory Synchronization/MS Mail for PCs
Lotus Directory Synchronization/Notes
Lotus Directory Synchronization/PROFS
Lotus Directory Synchronization/SYSM
Lotus Directory Synchronization/Toolkit
Lotus Directory Synchronization/Wang OFFICE
Lotus Gateway for CA-eMail+
Lotus Gateway for cc.Mail
Lotus Gateway for GroupWise
Lotus Gateway for Message Router
Lotus Gateway for MHS
Lotus Gateway for MS Mail for PC Networks
Lotus Gateway for MS Mail for PCs
Lotus Gateway for Notes
Lotus Gateway for VMSmail
Lotus Gateway for Wang OFFICE
Lotus Mail Monitor
Lotus mailFax
Lotus Messaging Switch
Lotus Messaging Switch Admin-By-Mail
Lotus Messaging Switch Application Toolkit/MVS

Figure A-8 (Page 2 of 2) Lotus Products

Program Product Name (Version/Release)
Lotus Messaging Switch Query-By-Mail
Lotus Messaging Switch Open Addressing Service
Lotus Pages
Lotus 1-2-3 (all current versions, OS/2 R2 1, DOS R4, Windows R5)
Notes Client Release 3 and Release 4 (all platforms)
Organizer 2 0
Remote Manager for Lotus Access Unit/cc:Mail for OS/2
Remote Manager for Lotus Access Unit/Notes for OS/2
ScreenCAM 2.1
SmartCenter for Windows
SmartCenter for OS/2
SmartCenter 96 for Win 95
Soft-Switch Central/MVS & VM
Soft-Switch Central Admin-By-Mail & Query-By-Mail
Soft-Switch Central Application Toolkit/MVS & VM
Soft-Switch Central Fax Gateway
Soft-Switch Central Open Addressing Service
Soft-Switch MAILbridge Server/DEC
Soft-Switch MAILbridge Server/HPDesk
Soft-Switch MAILbridge Server/MCI
Soft-Switch MAILbridge Server/MHS
Soft-Switch MAILbridge Server/SYSM
Soft-Switch MAILbridge Server/Wang OFFICE
Soft-Switch SNADS Gateway/Banyan
Soft-Switch SNADS Gateway/MS Mail for Appletalk Networks
Word Pro (all platforms)

Hardware

The hardware timers on the IBM S/390, AS/400, RISC/6000, and PowerPC machines are not sensitive to change of century. On a personal computer, the situation is more complicated.

IBM Personal Computer (PCs) - Hardware Timer Setting

In original PCs and XTs, the hardware did not have an internal real time clock. Therefore, the time and date had to be manually set each time the machine was booted. Beginning with the AT, personal computer systems maintain two dates, one date in the hardware component, and one in the operating system software (for example, DOS or Windows). The date in the operating system software is created by converting the date in the hardware component.

All IBM machines since the AT have a century byte in the hardware; it is used by the basic input/output system (BIOS) to determine the century. However, that century byte does not typically roll over at the end of 1999. You will have to reset it manually using either the Setup program (provided either on floppy disk, hard disk system partition, or via a BIOS program) or use the DATE command found in either DOS 3.3 or later or OS/2. This is the same procedure used to re-enter the date when the battery (which supports the timer) looses its charge and is replaced.

All new models of IBM PCs shipped in 1996 will automatically update the century byte.

For current IBM PCs and earlier models, some may require you use a command or a CMOS update utility to change centuries, however, some may not require any change.

The following tables list IBM PCs and PC Servers and indicate whether each model has manual or automatic 'century-byte-set' support.

Desktop PC Systems

Figure A-9 (Page 1 of 7). Desktop PC Systems - Internal Clock Setting

System	Model	Date Available	Century Byte Set Manually	Century Byte Set Automatically
PC	all		See Note	
XT	all		See Note	
XT286	all		X	
AT	all		X	
PS/1	all		X	
PS/2	all		X	
PS/55	all		X	
APTIVA	all before 1996		X	
2144-14P			X	
2144-16P			X	
2144-22P			X	

Figure A-9 (Page 2 of 7) Desktop PC Systems - Internal Clock Setting

System	Model	Date Available	Century Byte Set Manually	Century Byte Set Automatically
2144-24P			X	
2144-25P			X	
2144-27P			X	
2144-29P			X	
2144-66P			X	
2144-67P			X	
2144-82P			X	
2144-83P			X	
2144-86P			X	
2144-22F			X	
2144-24F			X	
2144-25F			X	
2144-27F			X	
2168-26P			X	
2168-62P			X	
2168-26F			X	
2168-62F			X	
2144-743			X	
2144-744			X	
2144-745			X	
2144-754			X	
2144-766			X	
2144-767			X	
2144-768			X	
2144-782			X	
2144-784			X	
2144-785			X	
2144-786			X	
2144-787			X	
2144-788			X	
2144-797			X	
2144-798			X	
2144-843			X	
2144-853			X	
2144-854			X	
2144-855			X	
2144-856			X	
2144-866			X	
2144-887			X	

Figure A-9 (Page 3 of 7) Desktop PC Systems - Internal Clock Setting

System	Model	Date Available	Century Byte Set Manually	Century Byte Set Automatically
2144-888			X	
2144-8K2			X	
2144-S82			X	
2144-S86			X	
2144-T82			X	
2168-755			X	
2168-756			X	
2168-789			X	
2168-792			X	
2168-857			X	
2168-889			X	
2144-H55			X	
2144-L55			X	
2144-H66			X	
2144-L66			X	
2144-K66			X	
2144-H77			X	
2144-L77			X	
2144-K77			X	
2144-KB1			X	
2144-KB2			X	
2144-H78			X	
2144-L78			X	
2144-K78			X	
2168-H89			X	
2168-L89			X	
2168-K89			X	
2144-H92			X	
2144-L92			X	
2144-K92			X	
2144-KB3			X	
2144-H67			X	
2144-L67			X	
2144-K67			X	
2144-H80			X	
2144-L80			X	
2144-K80			X	
2144-H81			X	
2144-L81			X	

Figure A-9 (Page 4 of 7) Desktop PC Systems - Internal Clock Setting

System	Model	Date Available	Century Byte Set Manually	Century Byte Set Automatically
2144-K81			X	
2144-KB4			X	
2144-MB4			X	
2144-KB5			X	
2144-MB5			X	
2144-H93			X	
2144-L93			X	
2144-K93			X	
2144-KB6			X	
2168-H90			X	
2168-L90			X	
2168-K90			X	
2144-X65			X	
2144-X70			X	
2144-X88			X	
2144-X79			X	
2144-X89			X	
2144-X90			X	
2144-Y65			X	
2144-Y70			X	
2144-Y78			X	
2144-Y84			X	
2144-Y88			X	
2144-Y79			X	
2144-Y89			X	
2144-Y90			X	
2144-26J			X	
2144-68J			X	
2144-27J			X	
2144-28J			X	
2168-62J			X	
2168-63J			X	
2168-64J			X	
2168-65J			X	
2144-P30		5/95	X	
2144-S15		5/95	X	
2144-LP0		6/95	X	
2144-LP1		6/95	X	
2144-LS1		8/95	X	

Figure A-9 (Page 5 of 7) Desktop PC Systems - Internal Clock Setting

System	Model	Date Available	Century Byte Set Manually	Century Byte Set Automatically
2144-KP1		8/95	X	
2144-LP2		6/95	X	
2144-LS2		8/95	X	
2144-KP2		8/95	X	
2144-XP1		6/95	X	
2144-XP2		6/95	X	
2144-YP1		6/95	X	
2144-YP2		6/95	X	
2144-70J		7/95	X	
2144-71J		7/95	X	
2144-KU0		8/95	X	
2144-KU1		8/95	X	
2144-M30		8/95	X	
2144-M40		8/95	X	
2144-900		8/95	X	
2144-910		8/95	X	
2144-911		8/95	X	
2144-930		8/95	X	
2144-LB0		8/95	X	
2144-KB0		8/95	X	
APTIVA	all after 1995			X
2144-C30		2H95	X	
2144-M31		2H95	X	
2144-C31		2H95	X	
2144-M35		2H95	X	
2168-M40		2H95	X	
2168-C40		2H95	X	
2168-M41		2H95	X	
2168-C41		2H95	X	
2144-M50		2H95	X	
2144-M51		2H95	X	
2144-C51		2H95	X	
2168-M52		2H95	X	
2168-M53		2H95	X	
2144-C53		2H95	X	
2168-M54		2H95	X	
2168-M55		2H95	X	
2168-M56		2H95	X	

Figure A-9 (Page 6 of 7) Desktop PC Systems - Internal Clock Setting

System	Model	Date Available	Century Byte Set Manually	Century Byte Set Automatically
2168-M57		2H95	X	
2168-M58		2H95	X	
2168-M60		2H95	X	
2168-M61		2H95	X	
2168-M62		2H95	X	
2168-M63		2H95	X	
2168-M70		2H95	X	
2168-M71		2H95	X	
2168-M72		2H95	X	
2168-M91		2H95	X	
2144-91W		2H95	X	
2144-914		2H95	X	
2144-921		2H95	X	
2144-92W		2H95	X	
2168-931		2H95	X	
2168-93W		2H95	X	
2168-932		2H95	X	
2168-9W2		2H95	X	
2168-934		2H95	X	
2168-935		2H95	X	
2144-937		2H95	X	
2144-941		2H95	X	
2144-94W		2H95	X	
2168-951		2H95	X	
2168-95W		2H95	X	
2168-964		2H95	X	
2168-96W		2H95	X	
2144-LL1		2H95	X	
2144-L10		2H95	X	
2144-K10		2H95	X	
2144-LL2		2H95	X	
2144-L20		2H95	X	
2144-K20		2H95	X	
2144-KU6		2H95	X	
2144-LL3		2H95	X	
2144-L30		2H95	X	
2144-K30		2H95	X	
2144-LL4		2H95	X	
2144-L40		2H95	X	

Figure A-9 (Page 7 of 7) Desktop PC Systems - Internal Clock Setting

System	Model	Date Available	Century Byte Set Menually	Century Byte Set Automatically
2144-K40		2H95	X	
2144-KU4		2H95	X	
2168-LL5		2H95	X	
2168-L50		2H95	X	
2168-K50		2H95	X	
2168-LL6		2H95	X	
2168-L60		2H95	X	
2168-K60		2H95	X	
2144-X10		2H95	X	
2144-Y10		2H95	X	
2144-Z10		2H95	X	
2144-X20		2H95	X	
2144-Y20		2H95	X	
2144-Z20		2H95	X	
2144-W30		2H95	X	
2144-X30		2H95	X	
2144-Y30		2H95	X	
2144-Z30		2H95	X	
2168-X40		2H95	X	
2168-Y40		2H95	X	
2168-Z40		2H95	X	
2144-N30		2H95	X	
2144-N40		2H95	X	
2144-N41		2H95	X	
2168-N50		2H95	X	
2168-N51		2H95	X	
2168-N60		2H95	X	
2168-N61		2H95	X	
2168-N71		2H95	X	

Note: No internal clock You must enter the date when you re-boot the system.

Commercial PC Desktop Systems

All models in Figure A-10 on page A-30 update the century byte automatically after you install the flash BIOS available from the Bulletin Board Service

Figure A-10. Commercial Desktop PC Systems - Internal Clock Setting

System	Model/ Type	Update Level	Date Avail- able	Century Byte Set Manually	Century Byte Set Automatically
6381		L8JT45A and higher	1996		X
6382/S		L6JT69A and higher	1996		X
6384/D		L6JT69A and higher	1996		X
6387/T		L6JT69A and higher	1996		X
6384 P60/D		1.00 06 and higher	1996		X
6472		LDJT71A and higher	1996		X
6482		LDJT71A and higher	1996		X
6484		LDJT71A and higher	1996		X
6492		LDJT71A and higher	1996		X
6494		LDJT71A and higher	1996		X
6571		LEJT62A and higher	1996		X
6573		LEJT62A and higher	1996		X
6575		N1JT63A and higher	1996		X
6581		LEJT62A and higher	1996		X
6583		LEJT62A and higher	1996		X
6585		N1JT63A and higher	1996		X
6875		N1JT63A and higher	1996		X
6876		N2JT38A and higher	1996		X
6885		N1JT63A and higher	1996		X
6886		N2JT38A and higher	1996		X

Figure A-11 Commercial Desktop PC Systems - Internal Clock Setting

System	Model/ Type	Date Available	Century Byte Set Manually	Century Byte Set Automatically
PS/V	2405/2410	10/92	X	

Figure A-11. Commercial Desktop PC Systems - Internal Clock Setting

System	Model/ Type	Date Available	Century Byte Set Manually	Century Byte Set Automatically
PS/V	2405	5/93	X	
PS/V	2405/2410	9/93	X	
PS/V Vision	2408	11/93	X	
PS/V Entry	2406	2/94	X	
PS/V MASTER	2411	5/94	X	
PC 750	6885	1/95	X	
PC 330	6571	3/95	X	
PC 720 ISA	6869	5/95	X	
OMNI-MC	6860-J0x/ JZG	01/95	X	
OMNI-MC	6860-J4G	01/95	X	
OMNI-MC(CR)	6860-Jxx	08/95	X	
5510	all		X	
5520	all		X	
5530	all		X	
5535	all		X	
5540	all		X	
5545	all		X	
5550	all		X	
5560	all		X	
5570	all		X	
5580	all		X	
6576		1996		X
6586		1996		X
6598	CXJ	12/95		X
6598	CXK	12/95		X
6598	CXX	12/95		X
6877				X
6887				X
8595	all		X	
9557	all		X	
9585	all		X	
9595	all		X	
550x	all		X	

Mobile PC Systems

Figure A-12 (Page 1 of 2). Mobil PC Systems - Internal Clock Setting

System	Model/ Type	Date Available	Century Byte Set Manually	Century Byte Set Automatically
700/C	9552	10/92	X	
700T	2521		X	
710T	2523	3/93	X	
720/C	9552	5/93	X	
730T	2524	5/94	X	
750C/Cs	9545	9/93	X	
750Ce	9545	11/93	X	
755C/Cs	9545	6/94	X	
755CE/CSE	9545	10/94	X	
755CD	9545	11/94	X	
755CX	9545	5/95	X	
755CV/CDV	9545	5/95	X	
760C	3546	10/95		X
701C	2630	3/95		X
500	2603		X	
510Cs	2604		X	
520CS			X	
530Cs	2605	5/95	X	
N45SL	2614		X	
300	2615		X	
340	2610	7/94	X	
340CSE	2610	2/95	X	
340CSE	2610	3/95	X	
350C	2618		X	
355C/Cs	2619	6/94	X	
360C/Cs/P	2620	5/94	X	
360CE/CSE/PE	2620C	0/94	X	
370C	9545	5/95	X	
345	2610	9/95		X
365C/CD/ CS/CSD	2625	11/95	X	
220	2432	7/93	X	
230Cs	2432	7/94	X	
PS/55	N51SX 8551	12/91	X	
PS/55	N51SLC 8551	5/92	X	
PS/55	N27SX 5527	4/92	X	

Figure A-12 (Page 2 of 2) Mobil PC Systems - Internal Clock Setting

System	Model/ Type	Date Available	Century Byte Set Manually	Century Byte Set Automatically
PS/55	N23SX 5523	10/91	X	
PS/55	23V 5523	10/92	X	
320	5523	5/93	X	
330Cs	5523	11/93	X	
550BJ	2437	2/93	X	
555BJ	2437	4/94	X	
365	2625	2H95		X

PC Servers

All new models of IBM Servers shipped in 1996 will automatically update the century byte.

Figure A-13 PC Servers - Internal Clock Setting

Server	Model/ Type	Date Available	Century Byte Set Manually	Century Byte Set Automatically
95	all	9/93		X
300	all	3/95	X	
320	all	6/95	X	
500	all	3/95		X

Appendix B. Year2000-Ready Solution Developer Products

Figure B-1 is a list of Year 2000-ready products provided by their respective Solution Developers.

IBM intends to update this list in subsequent releases of this publication. If you have a product that is Year 2000-ready, and you would like IBM to consider including it in this list, please follow the instructions in "Year2000-Ready Solution Developer Product and Tool Authorization" in the back of this book.

Disclaimer

IBM does not make any representations or endorsement of any of the following products. This information was provided by the Solution Developer and IBM has made no effort to independently verify the Year 2000-readiness of the products. The reader is responsible for checking with the individual Solution Developer to confirm the specific implementation of the Year2000 date transition. In any event, IBM shall not incur any liability by listing the following information.

Figure B-1 (Page 1 of 5). Year2000-Ready Solution Developer Products

Company	Solution Developer Product	Operating System	Solution Developer Product Version/Release
Advanced Software Technologies Company Ltd	ASTUTE	MVS	3.2
ALTAI Software	ZACK	MVS	3.20A
	ZACK	VSE	2.10A and up
	ZARA	MVS	1.1.0A
	ZEBB	MVS	2.20A
	ZEKE	MVS	4.1.C and up
	ZEKE	VSE	4.1.C and up
	ZELA	MVS	1.1.0A
American Software, Inc.	AMSOFT Accounts Payable	MVS, VSE	20
	AMSOFT Accounts Receivable	MVS, VSE	20
	AMSOFT Customer Order Processing	MVS, VSE	20
	AMSOFT DRP/2000 Series 3	MVS, VSE	20
	AMSOFT Forecaster/2000	MVS, VSE	20
	AMSOFT Inventory Control & Accounting (IC&A)	MVS, VSE	20
	AMSOFT MRP II	MVS, VSE	20
	AMSOFT Purchasing	MVS, VSE	20

© Copyright IBM Corp. 1995, 1996

Figure B-1 (Page 2 of 5). Year2000-Ready Solution Developer Products

Company	Solution Developer Product	Operating System	Solution Developer Product Version/Release
BlueLine Software, Inc	CACHE/VSE	VSE	2.1.1
	MAXBACK/VSE	VSE	2.2.5
	Multiprint/VM	VM	3.0.1
	Multiterm/VM	VM	5.2.2
	Multiterm/SNA	MVS	2.5.175
	Multiterm/SNA	VM	2.5.175
	Multiterm/SNA	VSE	2.5.175
	Netware for MVS	MVS	2.0.0
	Vital Signs/VM	VM	4.1.1, 4.1.4
	Vital Signs CICS	VSE	4.0.1
	Vital Signs VTAM [1]	MVS	4.0
	Vital Signs VTAM	VM	3.1.3
	Vital Signs VTAM	VSE	3.1.3
	VLOCK/VM	VM	3.03
CDB	CDB/Part Roll	MVS	all
	CDB/REXX	MVS	all
	CDB/Super Copy	MVS	all
	CDB/Super Load	MVS	all
	CDB/Super Reorg	MVS	all
	CDB/Super Restore	MVS	all
	CDB/Super Unload	MVS	all
Cole Software	XDC	MVS	X3.0
Compuware Corporation	DATA-XPERT	OS/2	1.0
	File-AID/MVS	MVS	8.0
Cybermation Inc	ESP Workload Manager	MVS	4.4.1
	ESP Workload Manager Extensions	MVS	4.4.1
	ESP Encore	MVS	2.1.3
	ESP Console Manager	MVS	1.1.1
	ESP Communication Server	MVS	1.1.1
	InfoServ	MVS	1.1.1
	ESP API	MVS	3.1.1
Data Base Architects, Inc	The OLR System [2]	MVS	all
The DBT Group, Inc	COMPRESS for DB2	MVS	any
	COMPRESS for IDMS	MVS	any
	COMPRESS for IMS	MVS	any
	DBT/Advisor for VSAM	MVS	any
	DBT/Control for VSAM	MVS	any
DTS Software, Inc	Allocation Control Center	MVS	1.1, 1.2, 2.1
	Space Recovery System	MVS	1.1, 1.2, 2.1
	Portal 2000	MVS	1.1
Davison Associates, Inc	MTS/CICS	MVS/CICS	1.1.7 or higher
	MTS/CICS	VSE/CICS	1.1.7 or higher

Figure B-1 (Page 3 of 5) Year2000-Ready Solution Developer Products

Company	**Solution Developer Product**	**Operating System**	**Solution Developer Product Version/Release**
Firesign Computer Company	Outbound ClientServer	MVS	3.2L
	Outbound ClientServer	VM	3.2L
	Outbound ClientServer	VSE	3.2L
	Outbound Host	MVS	3.2 L
	Outbound Host	VM	3.2.L
	Outbound Host	VSE	3.2.L
Gimpel Software	FlexeLint	VM	all
	FlexeLint	MVS	all
Information Management Company	TUXEDO	MVS	4.2.2
	Open TransPort	MVS	2.0.16
Information Technology Systems, Inc.	SoftSpy	MVS	all
	SoftSpy	VM	all
	SoftSpy	VSE	all
InfoTel Corporation	INFOCOPY for DB2	MVS	any
	INFOCYPHER	MVS	any
	INFOLOAD for DB2	MVS	any
	INFOPAK for MVS,VSAM, DB2, IMS DATACOM, IDMS	MVS	any
	INFORECOVERY for DB2	MVS	any
	INFORECOVERY for IMS	MVS	any
	INFOREORG for DB2	MVS	any
	INFOSCOPE for DB2	MVS	any
	INFOTRACE for DB2	MVS	any
Innovative DP Designs, Inc	IMS ToolKit		all
IntelliWare Systems, Inc	IntelliCONSOLE	VSE	all
	IntelliRESOURCE	VSE	all
KTS Informations-Systeme GmbH	DOMESTIC	MVS	95
	DOMESTIC	OS/2	95
Mazda Computer Corp. (Action Software Int.)	Change Action	MVS/XA, MVS/ESA	5.01
Merrill Consultants	MXG	all	all
META Health Technology	CHARMS	MVS	
	CHARMS	VSE	
North Ridge Software	The Network Director	MVS	4.1.0
	The Network Director	VSE	4.1.0
	The Network Director	VM	4.1.0
	The Network Center	VM	1.7.0
	The Network Center	MVS	1.7.0
OSYS AG	QPAC-Batch	MVS	all
	QPAC-Online	VSE	all
Panorama Software Corporation	Sunrise	MVS/ESA	N/A
Real Solutions	SAF	MVS	2.0
RSDSA	EOS	MVS	1.0
	RSD FOLDERS	MVS	3.1

Figure B-1 (Page 4 of 5). Year2000-Ready Solution Developer Products

Company	**Solution Developer Product**	**Operating System**	**Solution Developer Product Version/Release**
SAP AG, Germany	R/2	MVS	5.0
	R/2	VSE	5.0
Softworks, Inc	Capacity Plus for VSAM(VCP)	MVS/XA, MVS/ESA	any
	Catalog Solution	MVS/XA, MVS/ESA	any
	HSM Agent	MVS/XA, MVS/ESA	any
	Performance Solution[3]	MVS/XA, MVS/ESA	any
	Space Recovery Facility	MVS/XA, MVS/ESA	any
	TeraSAM	MVS/XA MVS/ESA	any
	VSAM Assist	MVS/XA, MVS/ESA	any
	VSAM Quick Index	MVS/XA, MVS/ESA	any
	VSAM Space Manager	MVS/XA, MVS/ESA	any
Stonehouse & Company	MONIES	MVS/ESA	
	MONIES	VSE/ESA	
Storage Technology Corporation	CAM	MVS, UNIX	1.0
	CommonParser	MVS, VM, UNIX	1.0
	EMV/MVS	MVS	1.0
	ExLM	MVS	2.1
	ExPR	PC,MVS	1.0.3
	IXFP	MVS,VM	
	LibraryStation	MVS	1.0
	MPPM	PC, VM, MVS, VSE	1.0
	MPST	PC, VM, MVS, VSE	1.0
	MVS/CSC	MVS	2.0
	NASF	MVS	1.1
	RCE	MVS	all
	Stager	MVS,UNIX	1.0
SuperWylbur Systems, Inc	SuperWylbur	MVS	4.3
Systemware	C/QUE	MVS	3.3 and up
	MPS	MVS	3.6
	X/PTR	MVS	3.3 and up
	JHS	MVS	3.3 and up
Thomson Software Products	NOMAD	VM,MVS	all

Figure B-1 (Page 5 of 5). Year2000-Ready Solution Developer Products

Company	**Solution Developer Product**	**Operating System**	**Solution Developer Product Version/Release**
Ton Beller GmbH	SIRON	AIX	94 1
	SIRON	MVS	94 1
	SIRON	OS/2	94 1
	SIRON	VM	94 1
	SIRON	VSE	94 1
Triangle Systems, Inc	IOF	MVS	IOF 7C

Notes.

[1] - Being renamed Vital Signs Vision Net

[2] - OLR System includes:

- Online Help
- Online Reference
- Online Note pad
- The OLR API

[3] - Performance Solution includes:

- I/O Plus for VSAM
- I/O Plus for xSAM
- HiperLoad Plus for VSAM
- VSAM I/O Plus

Appendix C. Bibliography

Non-IBM Publications

By Author

1. Appleton, Elaine L., *Call in the Cavalry Before 2000*, Datamation, 1 Jan 1996 p. 42
2. Arnold, Robert S., *Millennium Now: Solutions for Century Date Change Impact*, Application Development Trends, Jan 1995, pp. 60-67
3. Arnold, Robert S., *On Assurances of Year 2000 Compliance*, Inside DPMA, Data Processing Management Association, Jan 1996
4. Arnold, Dr. Robert S., *Resolving Year 2000 Problems in Legacy Software*, presentation at Software Quality Week, San Francisco, CA, 1 Jun 1995
5. Arnold, Robert S., *Tips for Performing A Software Inventory to Resolve the Year 2000 Problem*, Inside DPMA, Data Processing Management Association, Jan 1996
6. Barker, Paul, *End of Century Problem Looms As IS Remains Idle*, Computing Canada, Vol. 19, No. 23, 8 Nov 1994, pp. 1-4.
7. Barker, Paul, *Study Shows Impact of Year 2000*, Computing Canada, 94-47112, Vol. 19, No. 26, 20 Dec 1993, pp. 1-7
8. Bartholomew, Doug, *The Year 2000 Problem: Time's Running Out*, InformationWeek, 5 Feb 1996, p. 30
9. Baum, David, *Tool Up for 2000*, Datamation, 1 Jan 1996, p. 49
10. Baum, David, *Union Pacific Stays on Track for 2000*, Datamation, 1 Jan 1996, p. 39
11. Beizer, Boris, *Software Testing Techniques*,, Van Nostrand Reinhold Company, 1983
12. Betts, Mitch, *Desktops Veer Toward Year 2000 Crisis*, Computerworld, Vol. 28, No. 28, 11 Jul 1994, pp. 1-28
13. Betts, Mitch, *IBM Pledges Year 2000 Update in '96*, Computerworld, Vol. 29, No. 46, 13 Nov 1995, p. 79,87
14. Betts, Mitch, *Users Slow to Face Year 2000 Conversion*, Computerworld, Vol. 29, No. 15, 10 Apr 1995, p. 73
15. Bloom, Al, Managing System Development, Jan 1996.
16. Butler, Ken, *Can Your Computer System Handle the Change of the Century?*, Rough Notes, Vol.138, No. 9, Sep 1995, p. 28.
17. Celko, Joe, *Start Fixing Year 2000 Problems Now*, Datamation, 1 Jan 1996, p. 36
18. Cini, Al, *System Bug of the Apocalypse*, Internetwork, Jan 1995
19. Cohn, Michael B., *Dateline 1999: IS Pros Retire in Droves*, Computerworld, Vol. 29, No. 44, 30 Oct 1995, p. 37.

© Copyright IBM Corp. 1995, 1996

20 Cohn, Michael B., *No Need to Fear the Year 2000*, Computerworld, Vol. 28, No. 47, 21 Nov 1994, p. 35

21. Cox, Brian, *Year 2000 is Problematic for Systems*, National Underwriter (Life/Health/Financial Services), Vol. 98, No. 42, 17 Oct 1994, pp.7, 28

22 Cox, Brian, *Year 2000 is A Big Problem for Systems*, National Underwriter (Life/Health/Financial Services), Vol 98, No 47, 21 Nov 1994, pp 9, 18

23 de Jager, Peter, *Collateral Damage Will Surface*, Computer World Canada, Vol 12, No 1, 19 Jan 1995, p 6

24 de Jager, Peter, *Companies come clean*, Computerworld, Vol 29, No. 47, 20 Nov 1995, pp 97-100

25 de Jager, Peter, *Computer D-Day: Jan 1, 2000*, Toronto Star, 16 Jun 1994

26 de Jager, Peter, *Do your Systems have only 5 years to live?*, Best's Review (Life/Health), Vol. 96, No 2, May 1995, pp 88-90

27 de Jager, Peter, *Doomsday 2000*, Computerworld Vol. 27, No 36, 6 Sep 1993, pp. 105-109

28 de Jager, Peter, *Embrace the Future, let go of the Past*, Information Canada, Vol 19, No 12, Dec 1994, pp 5.

29 de Jager, Peter, *Feedback Highlights bad BIOS Blues*, Information Canada, Vol 20, No 2, Feb 1995

30 de Jager, Peter, *If you start now ... you just might make it* Computerworld, Vol 29, No. 47, 20 Nov. 1995, pp 97-100.

31 de Jager, Peter, *Only you can Decide*, Information Canada, Jun 1995

32 de Jager, Peter, *Start now or face 2000 Doom*, Datamation 96-73255, Vol. 41, No 9, 15 May 1995, p.92, 11.

33 de Jager, Peter, *Take a reporter out to lunch*, Datamation, 1 Jan 1996, p. 76

34 de Jager, Peter, *Time's a Wasting and The Clocks Keep on Tickin'*, Info Canada, Vol 18, No 5, May 1993, p 7.

35. de Jager, Peter, *2000 or Bust*, CA Magazine, Vol. 127, No 6, Aug 1994, pp 32-33

36 DeVoe, Deborah, *Still No Quick Fix to Year 2000 Glitch*, InfoWorld Magazine, Vol 17, No 48, 27 Nov 1995, p 46.

37 Dunn, Robert H., *Software Defect Removal*, McGraw-Hill, Inc., 1984

38 Edwards, John, *New Year's Evil*, CIO Magazine, 15, Dec 1995, p 82

39 Elms, Teresa, *Global Trends Shape Midrange for 2000*, System 3X/400, Vol 20, No 12, Dec 1992, pp. 44-50

40. Farber, Arnold and Rosemary LaChance, *Time Slips Away: Year 2000 is Closer Than You Think*, Enterprise Systems Journal, Feb 1996, p. 42.

41 Fine, Doug, *Companies Brace for Millennium*, Infoworld, 10 Apr 1995.

42 Furman, Jeff, Albert Marotta, and Cliff Candiotti, *Party When It's 1999*, Software Magazine, Apr 1995, pp 6-8

43 Furman, Jeff and Albert Marotta, *Year 2000 Denial*, Computerworld, Vol. 28, No. 43, 24 Oct 1994, pp. 70

44 Gartner Group Reports and Research Notes:

a Case, A., *System Date 2001, a Development Odyssey*, Gartner Group, Application Development & Management Strategies (ADM), Research Note, Strategic Planning SPA-210-835, 8 Feb 1993

b. Hall, B. and K. Schick, *Year 2000 Crisis: Estimating the Cost* Gartner Group, Applications Development and Management Strategies (ADM), Research Note, Key Issue Analysis KA-210-1262 28 Dec 1995

c. Hall, B and K Schick, *Year 2000 Crisis: Make Applications Compliant by YE1997*, Gartner Group, Applications Development and Management Strategies (ADM), Research Note, Strategic Planning SPA-215-1251 29 Nov 1995

d Jones, N , *PCs and the Year 2000*, Gartner Group, Applications Development and Management Strategies (ADM), Research Note, Strategic Planning, SPA-980-1278, 30 Jan 30, 1996

e Phelps, John, *Year 2000 - Known but Not Understood?*, Gartner Group, Large Computer Research Note, Strategic Planning, SPA-800-1734, 26 Jul 1995.

f Schick, K , *An RFP for the Year 2000 Date Change*, Gartner Group, Applications Development and Management Strategies (ADM), Research Note, Tutorials, TU-215-1147, 12 Apr 1995

g Schick, K., *INSPECT Legacy Applications for the Year 2000*, Garter Group, Application Development & Management Strategies (ADM), Research Note, Strategic Planning SPA-215-1146, 12 Apr 1995

h Schick, K., *Three Certainties. Death, Taxes and the Year 2000*, Gartner Group, Application Development & Management Strategies (ADM), Research Note, Strategic Planning SPA-215-1116, 25 Jan 1995

i Schick, K. and C Germann, *Building Year 2000 Contract Protection*, Gartner Group, Applications Development and Management Strategies (ADM), Research Note, Strategic Planning SPA-215-1252, 29 Nov 1995

j Schick, K and C Germann, *Date Change in the Year 2000 A Test of Asset Management*, Gartner Group, Applications Development and Management Strategies (ADM), Research Note, Strategic Planning, SPA-605-228, 25 July 1995

k Schick, K and C Germann, *Date Change in the Year 2000 A Test of Asset Management*, Gartner Group, Transition Strategies (TS), Research Note, Strategic Planning, SPA-LEG-162, 7 Aug 1995

l. Schick, K , *The Year 2000 Date Change Crisis Awareness* Gartner Group Continuous Services, Applications Development & Management Strategies (ADM), Strategic Analysis Report, R-215-130, 9 Feb 1996

45 Glass, Brett, *Turn of the Century Will Bring Out Flaws in Many Programs*, InfoWorld, Vol 17, No 25, 19 Jun 1995, p 42

46 Glass, Robert L , *The Date Wars Learning Management by Issue*, Information Systems Management, Vol 12, No 2, Spring 1995, pp 67-69

47 Goodwin, Bill, *Tick, Tick, Tick* , (newsletter), 2000AD, Inc , Brooklyn, NY

48 Greiner, Lynn, *Year 2000 a Time Bomb Ready to Explode*, Computing Canada, Vol. 21, No 14, 5 Jul 1995, p 11

49. Hoffman, Thomas, *1,418 days and counting - CA offers year 2000 date-change products*, Computerworld, 12 Feb 1996

50 Hayes, Brian, *Waiting for 01-01-00*, American Scientist, Volume 83, Jan-Feb 1995

51. Jones, David C., *Solving Year 2000 Problem Long, Costly Project*, National Underwriter (Life/Health/Financial Services), Vol. 99, No. 26, 26 Jun 1995, pp 9-24

52 Kilmar, and Wasserman, *Pros/Cons Procedure vs. Data*, American Programmer, Feb 1996

53 Lips, Michael, *Six-Digit Dates And The Century Change: Complex Problem. Simple Solution?*, Enterprise System Journal, Oct 1993, pp 68-69

54 Marcoccia, Lou, Managing System Development, Jan 1996

55 Martin, James, *Information Engineering, Book I Introduction,*, Prentice Hall, 1990

56 Martin, James, *Information Engineering, Book II Planning and Analysis*, Prentice Hall, 1990.

57 Martin, James, *Information Engineering, Book III Design and Construction*, Prentice Hall, 1990.

58 Martin, James and Joe Leben, *Strategic Information Planning Methodologies*, Second Edition, Prentice Hall, 1989

59 Martin, James and Carma McClure, *Structured Techniques The Basis for CASE*, Revised Edition, Prentice Hall, 1988.

60 Martin, Larry W., *Millennium Preparation*, Software Magazine, Vol 13, No 10, Jul 1993, pp 6-8

61. McCarthy, Vance, *Keep the Millennium Virus off Your Net*, Datamation, 1 Jan 1996, p. 55

62. McColgan, Declan, *Countdown to 2000*, Irish Computer, Feb 1996

63 McKendrick, Joseph, *A Once in a Century Crisis*, Midrange Systems, Vol. 8, No. 12, 30 Jun 1995, pp. 17-18.

64 McKendrick, Joseph, *The Year 2000: Still Way Off*, Midrange Systems, Vol. 8, No 19, 13 Oct 1995, p 3

65 Meador, C Lawrence, *Solving the Year 2000 Problem*, InformationWeek, 5 Feb 1996, p 44

66 Meall, Lesley., *The Century's Time Bomb*, Accountancy, Vol. 116, No 1228, Dec 1995, p 52

67. Miller, Edward and William E. Howden, *Tutorial: Software Testing & Validation Techniques*, Second Edition,

68 Neumann, Peter, G., *Computer Related Risks*, Addison-Wesley, ISBN 0-201-55805-X.

69 Olsen, Florence, *Govt. Urged to Perform Triage in Two-Digit Field Repairs*, Government Computer News, 18 Mar 1996.

70 Perry, William E., *A Structured Approach to Systems Testing*, QED Technical Publishing Group, 1983

71 Perry, William E., *Quality Assurance for Information Systems: Methods Tools, and Techniques*, QED Technical Publishing Group, 1991.

72 Peterson, DuWayne, *The Year 2000: Close Enough to be Dealt With*, SIM Network (Society for Information Management), Feb 1996

73 Peterson, Ivars, *Fatal Defect*, Times Books, ISBN 0-8129-2023-6

74. Poland, Tom, *Year 2000 Dilemma Creating Havoc with Software* National Underwriter (Life/Health/Financial Services), Vol 98, No 49, 5 Dec 1994, pp 38-39

75 Porter, Alan L , Frederic A Rossini, Stanley R Carpenter, A T Roper, Ronald W Larson, and Jeffrey S Tiller, *A Guidebook for Technology Assessment and Impact Analysis*, North Holland, 1980

76 Ross, Noah, *The End of the Century is Nearer Than You Think*, Application Development Trends, Apr 1995

77 Sager, Ira, *Glitch of the Millennium*, Business Week, No 3450, No 3450 13 Nov 1995, p 54

78 Scheier, Robert L., *Face up to it*, Computerworld, 25 Mar 1996, pp 83-84

79 Schick, Kevin, *The Year 2000 Date Crisis*, Government Finance Review, Vol 11, No 4, Aug 1995, pp 32-33

80 Schulmeyer, Gordon G , *Zero Defect Software*, McGraw-Hill, Inc , 1990

81. Schwartz, Susana, *Are Insurers Stuck in a Time Warp?*, Insurance & Technology, Vol. 20, No 5, May 1995, p 56

82. Stedman, Craig, *Utility charges to 2000*, Computerworld, 18 Feb 1996, p 71

83 Steward, Donald V , *Software Engineering with System Analysis and Design*, Brooks/Cole Publishing Company, 1987, IEEE Computer Society Press

84 Stitt, Martin, *Debugging: Creative Techniques and Tools for Software Repair* John Wiley & Sons, Inc., 1992.

85. Sullivan, R. Lee, *Ghosts in the Machines*, Forbes, 19 Jun 1995

86. Ulrich, William R., *10 pitfalls to avoid in year 2000 initiatives* Application Development Trends, Feb 1996

87 Vandercook, R. Gibbs, *Now Is the Time*, Systems Management, Vol 23, No 3 Mar 1995, p 66

88 Xenakis, John J, *The Year 2000 Surprise*, CFO The Magazine for Senior Financial Executives, Vol. 11, No 8, Aug 1995, p 22

89 Yourdon, Edward, *Modern Structured Analysis*, Yourdon Press, Prentice Hall, 1989

90 Zvegintzov, Nicholas, *The Year 2000 as racket and ruse*, American Programmmer, Feb 1996.

91 Zvegintzov, Nicholas, Managing System Development, Jan 1996

y Title

1 *Adpac Unveil Date Conversion Tool*, Computerworld, 10 Oct 1994, p 81

2. *ANSI Cobol Standard Modified For The Year 2000*, Information Week, 26 Feb 1990, p 19

3. ANSI X3.30-1985 (R1991) *Representation for Calendar Date and Ordinal Date for Information Interchange*.

4 ANSI X3 51-1994 *Information Systems – Representations of Universal Time, Local Time Differentials, and United States Time Zone References for Information Interchange*.

5. *Biting the Bullet*, Computerworld, 18 Mar 1996, p. 16.
6. *CAP Gemini Seeks To Solve Year 2000 Dilemma*, News 3x/400, Jun 95, p. 30
7. *Computer Horizons Establishes Mini-homepage on de Jager's Year 2000 Information Center Internet Home Page*, PR Newswire Association, 6 Nov 1995, p. 1106NY062.
8. *Firms Race Around the Clock to Avert "Millennium Bug" Bite*, Detroit News, 3 Dec 1995, p. C1
9. *For The Unwary, The New Millennium Could Bring Headaches*, Newsbytes News Network, Seattle, WA, 30 Mar 1995
10. *IBM Counts on Developers, VARs to Ring in Year 2000*, Computer Reseller News, CMP Publications, Inc., 13 Nov. 1995, p. 96.
11. *IBM Plans Year 2000 Upgrades*, InformationWeek, CMP Publications, Inc., 13 Nov. 1995, p. 30
12. *IBM Prepares for Year 2000 Date*, Newsbytes News Network, 2 Nov 1995
13. *IT Begins to Face Year 2000 PC Data Problem*, Newbytes News Network, 31 Oct. 1995.
14. ISO 8601:1988 *Data elements and interchange formats – Information interchange – Representation of dates and times*, Technical Committee ISO/TC 154.
15. *Keane, VIASOFT Announce Agreement to Provide Complete Year 2000 Solution*, Business Wire, 4 Nov. 1995, p11061123
16. *Mainframe Time Bomb: 01-01-00*, Fortune Magazine, 6 Mar 1995, p. 24
17. *Micro Focus Offers Year 2000 Fixer*, Computerworld, 31 Jul 1995, p. 8.
18. *Millennium could bring computer chaos*, Southam News organization as reported in The Daily News, Halifax, N.S., Canada, Feb 1996
19. *New OS/390 handles year 2000*, Computerworld, 19 Feb 1996,
20. *Ottawa Confident of Escaping Year 2000 Rollover Debacle*, Technology in Government, Feb 1996
21. *Pain or Gain in the Year 2000?*, Computer Business Review, Mar 1996
22. *SEEC Ship COBOL Tool Upgrade*, PC Week, 21 Nov 94, p. 31
23. *Seec Upgrades Analysis & Maintenance*, Software Magazine, Feb 1995, p. 93
24. *Service for Year 2000 Year Date Processing of Programs Software*, New Technology Japan, Feb. 1995, p. 24.
25. *Technology Problem Of The Century*, Bank Technology News, Faulkner & Gray, Inc., Jun. 1995, p. 10
26. *Time running out for Year 2000 transition*, Network World Canada, Vol. 6, No. 2, 2 Feb 2 1996
27. *Troubled Time*, Associated Press, Denver Post, (and various other USA newspapers), 25 May 1995.
28. *Users Invest in Millennium Changes*, Xephon 1995, Insight IS, pp. 12-13.
29. *Vendors Focus On Year 2000 Conversion Issue*, Computerworld, 10 Apr 1995, p. 73.
30. *VIASOFT and Litton Computer Services Enter into a Strategic Alliance*, Business Wire, 1 Dec 1995.

31. *Viasoft's Unveil Code Decipher Software*, Computerworld, 15 Aug 1994, p 16 *Worried About 2000? Group Shares The Fretting*, Investor's Business Daily, 7 Feb 1996, p A6

32. *Year 2000: Keeping Doomsday Costs Down*, Enterprise Data Center Strategies, Meta Group, Inc., File No 431, 29 Dec 1994

33. *The Year 2000 Doesn't Compute*, Electronic Buyers' News, CMP Publications, Inc., 2 Jan 1996, p 2

Electronic (Internet/World-Wide Web) Documentation

1 ANSI Online Home Page (access to ANSI catalog and standards documents)

`http://www.ansi.org`

You can order both ANSI and ISO standards from ANSI by calling (212) 642-4900 between 8 45 a m and 4 45 p m eastern time (Note this is New York City, New York, USA)

2 Cable News Network - *The year 2000 does not compute*, CNN's Headline News, 7 January 1996

`http://www.cnn.com/TECH/9601/2000/index.html`

3 *Year 2000 Fix It Now!*, Datamation, Special Report Year 2000, 1 January 1996

`http://www.datamation.com/PlugIn/issues/1996/jan1/FEATURES.html`

4 *The Globe and Mail* (Canada's National Newspaper)

`http://www.globeandmail.ca/Cyber-17.html`

5 IBM Software Page

To access this document (*The Year 2000 and 2-Digit Dates A Guide for Planning and Implementation*) through the Internet's World Wide Web, visit the IBM Software Page at

`http://www.software.ibm.com/`

or IBM System/390 Home Page at

`http://www.s390.ibm.com/stories/tran2000.html`

Select an appropriate version of this document to download the file for printing to your printer, the following formats are available

ASCII text
Browseable BookManager
List3820
PostScript
UNIX compressed
Tersed listps
Tersed list3820

You can also access this document through an anonymous ftp whose URL is

`ftp://lscftp.kgn.ibm.com/pub/year2000/y2kpaper.`**`format_type`**
`where` **`format_type`** `is specific to one of the print formats`
`listed above.`

6 IBM DFSORT/MVS Home Page

To access DFSORT/MVS URL and read about DFSORT's Year2000 enhancements

`http://www.storage.ibm.com/storage/software/sort/srtmhome.htm`

You can also download PostScript and/or text versions of DFSORT's year2000 enhancements (name sortpy2k zip) via anonymous ftp.

`ftp://lscftp.kgn.ibm.com/pub/mvs/docs/`

7 Information Technology Association of America (ITAA) White paper available at

`http://www.itaa.org`

8 InformationWeek Year 2000 chat session

http://www.cmp.com/cgi-bin/techtalk/2000

9 InformationWeek web site:

http://techweb.cmp.com/techweb/iw/current/

10. National Bulletin Board for Year 2000 Home Page

http://www.it2000.com

11 ISO Online Home Page (access to ISO catalog and standards documents)

http://www.iso.ch

You can order both ASO and ANSI standards from ANSI by calling (212) 642-4900 between 8:45 a.m and 4:45 p.m. eastern time (Note this is New York City, New York, USA.)

12 Project 21st Century Home Page at

http://www.webhelp.com/future.htm

13 Royal Greenwich Observatory (RGO) (2000 AD, Leap Years,...)

http://www.ast.cam.ac.uk/RGO/leaflets/

14 *TechLink Newsletter* from CMP's TechWeb - The Technology Information Source, 8 Feb 1996

http://techweb.cmp.com

- Cover story location
  http://techweb.cmp.com/iw/565/65mtyr2.htm
- Open Labs location:
  http://techweb.cmp.com/iw/565/65olyr2.htm
- Threaded chat location
  http://www.cmp.com/cgi-bin/techtalk/2000

15 Texas Tech University Health Sciences web site (links to magazine articles and Year2000 Home Page and an archive of the Year 2000 newsgroup's daily digests) at

http://www.ttuhsc.edu/pages/year2000/ttuy2k.htm

16 USA Today Web site at

http://www.usatoday.com/news/comment/colmane.htm

XPRESS Software, Inc Home Page

http://www.xpsoft.com/

17 Year 2000 Frequently Asked Questions (FAQ) access

ftp://www.year2000.com/pub/year2000/y2kfaq.txt

18 The Year 2000 Home Page (facilitator Peter de Jager)

http://www.year2000.com/

To subscribe to the year2000 general forum, send

SUBSCRIBE YEAR2000

in the body of a message to listmanager@hookup net

19 *Year 2000 News* (E-zine) (facilitator Dr Robert S. Arnold)

To subscribe, send E-mail note to
new2000-request@andrew.cais.com

`enter SUBSCRIBE`

on the `SUBJECT` line

20 Year 2000 Software Control

`http://www.iac.honeywell.com/Pub/Tech/CM/CMTools.html`

21 Year 2000 Technical Audit Center

`http://www.auditserve.com.yr2000/countdown.cgi?2000,1,0,XX,XX,XX`

22 Year 2000 Web Links

`http://www.club.innet.be/~janjedsp/y2k.html`

Audios and Videos

1. *Millennium: A Billion Dollar Software Crisis*, Computer Channel, Inc., Presenters Dr Howard Rubin (Hunter College) and Jim Woodward (CAP Gemini America) Contact

    Computer Channel
    6801 Jericho Turnpike
    Syosset, New York 11791
    telephone: (516) 921-5170

2. *Millennium Madness*, panel discussion at 7th Annual Data Administration Management Association International Symposium, panel members Larry English, Clive Finkelstein, Ron Ross, and John Zachman, Vancouver, Canada, May, 1995 To order the VHS format video

    DAMA International
    P.O. Box 6096
    Chesterfield, Missouri 63006-6096

3. *Systems for the year 2000 The Upcoming Data Crisis*, Peter de Jager, 1995 Systems Forum & Exhibit (LOMA), 1995. To order the 60-minute audio tape

    Peter de Jager
    22 Marchbank Cres.
    Brampton, Ontario L6S 3B1
    CANADA

IBM Publications

By Author

1 Frawley, M D and H L Stuck, *Year 2000 Clean*, Version 1.3, Document Number TR 29 1601, 1994

2 Hillier, Richard, *Year 2000 Working Document*, Document Number YE-01, IBM Euroco

3 Ohms, Bruce G., *Computer Processing of Dates Outside the Twentieth Century*, IBM System Journal, Vol 25, No. 2, 1986, pp.244, G321-5274

Standard Publications

1 *CICS/ESA (Version 4 Release 1): System Definition Guide*, SC33-1164

2 *DB2 SQL Reference*, SC26-4890

3 *IBM High Level Assembler/MVS & VM & VSE: Language Reference (Release 1)*, SC26-4940

4 *IBM VSE/Enterprise Systems Architecture (Version 1 Release 3): System Macros Reference*, SC33-6516

5 *IMS/ESA Version 4: Utilities Reference: Database*, SC26-4627

6 *OS/390 V1R2.0 MVS Assembler Services Guide*, GC28-1762

7 *OS/390 V1R2.0 MVS System Commands*, GC28-1781

8 *OS/390 V1R2.0 MVS Assembler Services Reference*, GC28-1910

9 *National Language Design Guide Volume 2: National Language Support Reference Manual*, Fourth Edition, IBM Corp., 1994

10 *ESA/390 Principles of Operations*, SA22-7201

11 *Systems Application Architecture: Common Programming Interface C Reference - Level 2*, SC09-1308

12 *Systems Application Architecture: Common Programming Interface COBOL Level 2 Reference*, SC26-4726

13 *Systems Application Architecture: Common Programming Interface Language Environment Reference*, SC26-4970

14 *Systems Application Architecture: Common Programming Interface PL/I Reference*, SC26-4381

15 *Systems Application Architecture: Common Programming Interface REXX Level 2 Reference*, SC24-5549

16 *Virtual Machine/ Enterprise Systems Architecture: CP Command CP Command and Utility (Release 2)*, SC24-5519

17 *VS COBOL II: Application Programming Language Reference*, GC26-4047

18 *VS FORTRAN (Version 2 Release 5): Language and Library Reference*, SC26-4221

19 *Why Migrate to COBOL/370 and LE/370?*, COBMGVAL PACKAGE on MKTTOOLS

Appendix D. Glossary

This glossary defines data processing and communication terms used in this publication.

Numerics

2-digit-year format. A format that provides a year date as two digits only to represent a year within a specific century The two high-order digits of the year are truncated; for example 1995 is represented as 95, and 00 represents years ..1800, 1900, 2000....

4-digit-year format A format that provides a year date as four digits: the two high-order digits represent the century and the two low-order digits represent the year within the century For example, 1995 represents the year 1995; 2095 represents the year 2095.

C

CCYY format A 4-digit-year format that uses two century digits (CC) to indicate the century and two year digits (YY) to indicate the year within the century The CC representation is provided as either the actual century digits (for example, 18, 19, or 20) or as a encoded value (for example, as 00 to represent 19, 01 to represent 20 as in, 0095 represents the year 1995 and 0195 represents the year 2095.)

century A period of 100 consecutive years

Although IBM recognizes that the 21st century begins at 0000 hrs, 1 January 2001, for purposes of this document, we are defining the 20th—21st century boundary to be between 2400 hrs, 31 December 1999 and 0000 hrs, 1 January 2000 This allows a discussion of the 21st century to include all dates with a 20yy format inclusive of the year 2000 Hence, the year 2100 is likewise relegated to the 22nd century

century byte The high order byte of a field used to contain the two high order digits of a 4-digit year (For example, 19 in 1995, 20 in 2000 and 2001)

cosmetic Referring to a 2-digit-year date that is viewed by human eyes only, such as a print date on hardcopy output or a date on a selection panel Because it is neither read nor further processed by a program you might be able to exclude its modification from your year2000 work effort.

E

external side The receiver of a data entity Used in this document to mean a module or routine that accepts a 2- or 4-digit-date format entity for further processing from another module or routine.

F

fixed window A technique to determine the century (high-order digits) of a year when represented by two digits The 2-digit year is compared against a hardcoded threshold The century designation is limited to a 100-year range spanning only two centuries. For example, assume the threshold is 60, then if the 2-digit year is ≥60, the year is in the 20th century; if the 2-digit year is <60, the year is in the 21st century.

G

Gregorian calendar Today's general-use calendar of 12 months and 365 days that employs the current leap year algorithm (refer to **Leap year** below).

I

internal side The creator or manipulator of a data entity. Used in this document to mean a module or routine that externalizes a 2- or 4-digit-year format entity to another module or routine

J

Julian date A date in the format YYDDD A date format that contains the year in positions 1 and 2, and the day in positions 3 through 5 The day is represented as 1 through 366, right adjusted, padded with zeroes on the left

© Copyright IBM Corp. 1995, 1996

L

Leap year A year either evenly divisible by 400 or evenly divisible by 4 and not evenly divisible by 100 For example, the year 1900 was not a leap year but the year 2000 is a leap year

Lilian date The number of days since 14 October 1582 15 October 1582 is Lilian day 1, 16 October 1582 is Lilian day 2, and so on (Named for Aloysius Lilius (an advisor to Pope Gregory XIII) who, together with his brother, constructed the current Gregorian calendar)

R

rolling window Synonymous with **sliding window**

S

sliding window A technique to determine the century (high-order digits) of a year when represented by two digits. The user specifies the number of years (both past and future) within a 100-year window spanning two centuries For example assume the window is set at 19 future years (1996-2014) and 80 past years (1915-1994) Dates in the range 00-14 (inclusive) are designated 21st century dates because they fall into the future window Dates in the range 15-99 (inclusive) fall into the 20th century

T

TRANSFORMATION 2000 IBM's Information Systems Solution Corporation (ISSC) comprehensive set of solutions that takes into account applications, systems software and hardware in both a centralized and/or distributed environment

Y

Year2000 challenge The work-effort required to complete a Year2000 transition, to include planning, identifying, reformatting, testing, and migrating phases

Year2000 problem The potential problems and its variations that might be encountered in any level of computer hardware or software from microcode to application programs, files, and databases that need to correctly interpret year-date data represented in 2-digit-year format

Year2000 ready The capability of a software program to correctly interpret and manipulate year-date data outside the 1900-1999 year range and produce valid arithmetic results

Year2000 transition The process of revising all programming entities (programs, databases, and so on) to correctly process date data outside the range 1900-1999

YY format Synonymous with **2-digit-year format**

YYYY format Synonymous with **4-digit-year format** and a subset of **CCYY format**

Numerics

20th century The period of time 0000.00 hrs 1 January 1901 through 2400 00 hrs 31 December 2000

21st century The period of time 0000 00 hrs 1 January 2001 through 2400.00 hrs 31 December 2100

Index

Special Characters

- *DATE 7-43
- *DAY 7-43
- *MONTH 7-43
- *YEAR 7-43

Numerics

- 2-digit-year format
  - definition D-1
- 20th century
  - definition D-2
- 21st century
  - definition D-2
- 4-digit-year format
  - definition D-1

A

- ADDDUR (add duration) operation code 7-44
- AIX platform
  - year2000 readiness A-16
- arithmetic operations using OPNQRYF command
  - date 7-56
  - time 7-57
  - timestamp 7-58
- audio documentation
  - access C-11

B

- bibliography C-1
- bridge programs
  - used as a conversion tool 4-10

C

- calculations
  - incorrect 1-2
- CCYY format
  - definition D-1
- century
  - definition D-1
- century byte
  - definition D-1
- COBOL
  - tools 7-7, 7-29
- collating
  - incorrect 1-2
- commercial desktop PCs
  - setting hardware timer A-30
- cosemtic
  - definition D-1
- cosmetic date 4-1

D

- data exposure type
  - solution considerations 4-9
- data sharing 3-4
- date
  - arithmetic operations 7-50
  - arithmetic using OPNQRYF command 7-56
  - comparison using OPNQRYF command 7-54
  - cosmetic 4-1
  - duration 7-49, 7-55
- date and time
  - arithmetic operations 7-50—7-53
- date data field 7-40
- date format
  - for DB2/400 SQL 7-48
  - specifying current value 7-49
- date value
  - for DB2/400 SQL 7-48
- dates
  - used as special values 1-2
- debugging 5-1
- definition
  - 2-digit-year format D-1
  - 20th century D-2
  - 21st century D-2
  - 4-digit-year format D-1
  - CCYY format D-1
  - century D-1
  - century byte D-1
  - cosmetic D-1
  - external side D-1
  - fixed window D-1
  - Gregorian calendar D-1
  - internal side D-1
  - Julian date D-1
  - leap year D-2
  - Lilian date D-2
  - rolling window D-2
  - sliding window D-2
  - TRANSFORMATION 2000 D-2
  - year2000 challenge D-2
  - year2000 problem D-2
  - year2000 ready D-2
  - year2000 transition D-2
  - YY format D-2
  - YYYY format D-2
- desktop PCs
  - setting hardware timer A-24
- desktop PCs (commercial)
  - setting hardware timer A-30
- documentation
  - access C-8

© Copyright IBM Corp 1995, 1996

duration
- date 7-49
- labeled 7-49
- time 7-50
- timestamp 7-50

duration (date, time, and timestamp) 7-55

E

edit, date 7-44
electronic documentation C-8
examples
- CURRENT DATE 7-49
- CURRENT TIMEZONE 7-49
- special register 7-49

exposures
- locating 3-1

exposures classification 1-2
expression
- date and time operands 7-49

external side
- definition D-1

EXTRCT (extract date/time) operation code 7-44

F

fixed window
- definition D-1

G

glossary D-1
Gregorian calendar
- definition D-1

guidelines
- for using reformatting techniques 4-12

H

hardware A-24
- AS/400 A-24
- PowerPC A-24
- RISC/6000 A-24
- S/390 A-24

hardware timer
- setting for personal computers A-24

I

IBM consulting and services
IBM products A-1
IBM Year2000 solution
- assessment and strategy 8-1
- detailed analysis & planning 8-1
- implementation 8-2
- TRANSFORMATION 2000 8-1
- Year 2000 CLEAN MANAGEMENT 8-2

identifying 2-digit years
- approaches 3-1

identifying 2-digit years *(continued)*
- locating direct references 3-1
- locating indirect references 3-1
- using a test system 3-1

impact
- severity category 2-3

integrity 1-2
internal format
- default formats 7-40
- definition 7-40

internal side
- definition D-1

internet
- documentation access
  - to documentation C-8
  - to forums C-8

introduction
- to Year2000 transition 1-1

J

Julian date
- definition D-1

L

labeled duration 7-49, 7-55
leap year
- calculation 1-3
- definition D-2

Lilian date
- definition D-2

locating 2-digit years 3-1
locating exposures 3-1
Lotus Products
- year2000 readiness A-22

M

misconceptions
- of the year2000 challenge 1-1

mobile PCs
- setting hardware timer A-33

MVS platform
- year2000 readiness A-3

O

Open Query File (OPNQRYF) command
- using
  - date, time, and timestamp arithmetic 7-54
  - date, time, and timestamp comparison 7-54

operand
- date and time 7-49

OPNQRYF (Open Query File) command
- using
  - date, time, and timestamp arithmetic 7-54
  - date, time, and timestamp comparison 7-54

- **OS/2 platform**
  - year2000 readiness A-20
- **OS/400 platform**
  - year2000 readiness A-13

P

- **PC servers**
  - setting hardware timer A-35
- **PCs**
  - hardware timer setting A-24
- **personal computer**
  - hardware timer setting A-24
- **phases of testing**
  - debugging 5-1
- **PL/I**
  - tools 7-18, 7-29
- **planning**
  - considerations 2-2
  - to resolve exposures 2-1

R

- **recommendations**
  - for using reformatting techniques 4-12
- **references** C-1
- **reformatting**
  - year-date notation 4-1
- **reformatting techniques**
  - guidelines 4-12
- **rolling window**
  - definition D-2

S

- **sequence**
  - incorrect 1-2
- **sharing**
  - data 3-4
- **sliding window**
  - definition D-2
- **solution considerations**
  - to data exposure type 4-9
- **Solution Developer products** B-1
- **solutions**
  - for reformatting year notation
    - encoding 4-6
    - externalize 4-digit format 4-1
    - fixed window 4-3
    - sliding window 4-3
    - use common service routine 4-9
- **special value dates** 1-2
- **special words** 7-43
- **standards**
  - ANSI 4-12
    - access via www C-8
  - ISO 4-12
- **statements**
  - for DB2/400 SQL
    - date value 7-48
- **statements** *(continued)*
  - for DB2/400 SQL *(continued)*
    - time value 7-48
    - timestamp value 7-48
- **SUBDUR (subtract duration) operation code** 7-44

T

- **technique**
  - for reformatting year notation 4-1
    - compress 4-6
    - encoding 4-6
    - externalize 4-digit format 4-1
    - fixed window 4-3
    - sliding window 4-3
    - use common service routine 4-9
- **TEST (test date/time/timestamp) operation code** 7-44
- **testing** 5-1
  - a function's implementation 5-1
  - acceptance testing 5-1
  - end-user requirements 5-2
  - error handling 5-3
  - functional 5-2
  - integration testing 5-1
  - intersystem 5-3
  - manual support 5-3
  - operations 5-1
  - parallel 5-3
  - program validation 5-1
  - program verification 5-1
  - recovery 5-2
  - requirements 5-2
  - specifications 5-2
  - stress 5-1
  - structural 5-1
  - system testing 5-1
  - unit testing 5-1
- **time**
  - arithmetic operations 7-52
  - arithmetic using OPNQRYF command 7-57
  - comparison using OPNQRYF command 7-54
  - duration 7-50, 7-55
- **time data field** 7-42
- **time format**
  - for DB2/400 SQL 7-48
  - specifying current value 7-49
- **time value**
  - for DB2/400 SQL 7-48
- **timer**
  - desktop PCs A-24
  - desktop PCs (commercial) A-30
  - mobile PCs A-33
  - PC server A-35
- **timers**
  - setting for PCs A-24
- **timestamp**
  - arithmetic operations 7-53
  - arithmetic using OPNQRYF command 7-58
  - comparison using OPNQRYF command 7-54

timestamp *(continued)*
  duration 7-50, 7-55
timestamp data field 7-42
timestamp format
  for DB2/400 SQL 7-48
  specifying current value 7-49
timestamp value
tools 7-1
  characteristics 7-1
  COBOL 7-7, 7-29
  environment 7-1
  for code editing 7-4
  for code generation 7-5
  for code restructuring 7-4
  for hardware 7-1
  for impact analysis 7-2
  for program level analysis 7-3
  for project management 7-3
  for software 7-1
  necessary features 7-1
  PL/I 7-18, 7-29
  prerequisite hardware 7-1
  prerequisite software 7-1
  to analyze complexity 7-2
  to analyze consistency 7-4
  to analyze data flow 7-3
  to analyze databases 7-3
  to analyze interfaces 7-4
  to analyze logic 7-4
  to analyze metrics 7-3
  to analyze standards 7-4
  to analyze tests 7-5
  to automate testing 7-5
  to browse code 7-4
  to compare programs 7-4
  to create standard date subroutines 7-5
  to cross reference 7-4
  to diagram data structure 7-3
  to diagram decomposition 7-3
  to diagram logic structure 7-3
  to diagram relationships 7-3
  to expand fields 7-4
  to find dates 7-4
  to generate code 7-5
  to generate database code 7-5
  to generate dialog 7-5
  to generate reports 7-5
  to generate tests 7-6
  to inventory software 7-3
  to modularize code 7-5
  to organize data 7-6
  to paint screens 7-5
  to simulate system behavior 7-5
  to slice programs 7-4
  to test drivers 7-6
  to trace requirements 7-5
  to track changes 7-3
  types 7-2
TPF platform
  year2000 readiness A-8
TRANSFORMATION 2000
  definition D-2
  IBM Year2000 solution 8-1

U
UDATE 7-43
UDAY 7-43
UMONTH 7-43
UYEAR 7-43

V
video documentation
  access C-11
VM platform
  year2000 readiness A-11
VSE/ESA platform
  year2000 readiness A-9

W
world wide web (www) C-8

Y
year-date notation
  reformatting 4-1
Year2000
  exposure classification 1-2
  introduction 1-1
  problem scope 1-3
year2000 challenge
  definition D-2
year2000 problem
  definition D-2
year2000 readiness
  AIX platform A-16
  Lotus products A-22
  MVS platform A-3
  OS/2 platform A-20
  OS/400 platform A-13
  TPF platform A-8
  VM platform A-11
  VSE/ESA platform A-9
year2000 ready
  definition D-2
  IBM products A-1
  Solution Developer products B-1
year2000 transition
  definition D-2
year2000-related publications C-1
YY format
  definition D-2
YYYY format
  definition D-2

Year2000-Ready Solution Developer Product and Tool Authorization

If your product(s) is year2000-ready or you market tools designed to assist others attain year2000 readiness, and you would like IBM to consider including that product(s) in this publication, please supply the following information For products not currently "Year2000 ready," please notify us as they become so Please return to

Year2000 Solution Developer Liaison
IBM Corporation
Dept MNRA, Bldg 005-3, M/S P136
522 South Road
Poughkeepsie, NY 12601-5400
Fax (914) 432-9418

Your Product's Name *	Your Product's Version/Release	IBM Platform (and Version/Release) your product supports	Technical Contact (name, tele/e-mail)	Year2000 Tool? (Y / N)

* Please indicate any trademarks

I hereby authorize IBM to reference my company's name, address and phone number, along with the products and registered trademarks in conjunction with IBM's marketing activities This information may be edited as IBM deems appropriate, and may be included in, but not limited to, IBM's publication on the year 2000 date format transition, *The Year 2000 and 2-Digit Dates A Guide for Planning and Implementation*

Company ____

Address ____

____

Phone Number (___)-_______-_______ FAX Number (___)-_______-_______

E-mail address ____________________________

WWW Home Page ____________________________

Name ______________ Title ____________________________

Signature ____

© Copyright IBM Corp 1995, 1996

Readers' Comments — We'd Like to Hear from You

Third Edition

The Year 2000 and 2-Digit Dates:
A Guide for Planning and Implementation

Publication No. GC28-1251-02

You may use this form to communicate your comments about this publication, its organization, or subject matter, with the understanding that IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you. Your comments will be sent to the author's department for whatever review and action, if any, are deemed appropriate.

Note: Copies of IBM publications are not stocked at the location to which this form is addressed. Please direct any requests for copies of publications, or for assistance in using your IBM system, to your IBM representative or to the IBM branch office serving your locality.

Today's date ________________________________

What is your occupation?

Newsletter number of latest Technical Newsletter (if any) concerning this publication:

How did you use this publication?

[ ] As an introduction
[ ] As a reference manual
[ ] For another purpose (explain)
[ ] As a text (student)
[ ] As a text (instructor)

____

____

Is there anything you especially like or dislike about the organization, presentation, or writing in this manual? Helpful comments include general usefulness of the book, possible additions, deletions, and clarifications, specific errors and omissions.

Page Number Comment:

____
Name

____
Company or Organization

____
Phone No

____
Address

____

____

Communicating Your Comments to IBM

Third Edition

The Year 2000 and 2-Digit Dates
A Guide for Planning and Implementation
Publication No. GC28-1251-02

If you especially like or dislike anything about this book, please use one of the methods listed below to send your comments to IBM. Whichever method you choose, make sure you send your name, address, and telephone number if you would like a reply.

Feel free to comment on specific errors or omissions, accuracy, organization, subject matter, or completeness of this book. However, the comments you send should pertain to only the information in this manual and the way in which the information is presented. To request additional publications, or to ask questions or make comments about the functions of IBM products or systems, you should talk to your IBM representative or to your IBM authorized remarketer.

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you.

If you are mailing a readers' comment form (RCF) from a country other than the United States, you can give the RCF to the local IBM branch office or IBM representative for postage-paid mailing.

- If you prefer to send comments by mail, use the RCF at the back of this book.
- If you prefer to send comments by FAX, use this number:

  FAX: (International Access Code) + 1 + 914 + 432-9405
- If you prefer to send comments electronically, use this network ID:
  - IBMLink (United States customers only): KGNVMC(MHVRCFS)
  - IBM Mail Exchange: USIB6TC9 at IBMMAIL
  - Internet e-mail: mhvrcfs@vnet.ibm.com
  - World Wide Web: http://www.s390.ibm.com/os390

Make sure to include the following in your note:

- Title and publication number of this book
- Page number or topic to which your comment applies

Readers' Comments — We'd Like to Hear from You
GC28-1251-02

IBM®

Cut or Fold
Along Line

Fold and Tape

Please do not staple

Fold and Tape

NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST-CLASS MAIL PERMIT NO 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

IBM Corporation
Department 55JA, Mail Station P384
522 South Road
Poughkeepsie NY 12601-5400

Fold and Tape

Please do not staple

Fold and Tape

Cut or Fold
Along Line

Printed in U.S.A.

GC28-1251-02

Chairwoman MORELLA. Thank you very much for your excellent testimony.

Mr. Sokol.

STATEMENT OF MARC SOKOL, VICE PRESIDENT OF ADVANCED TECHNOLOGIES, COMPUTER ASSOCIATES INTERNATIONAL, INCORPORATED, ISLANDIA, NEW YORK

Mr. SOKOL. Madam Chairwoman and members of the Committee, it's a pleasure to testify on behalf of Computer Associates before your Committee on the ways that computer software programs can help expedite the correction of date processing flaws in computer software applications. The millennium date change can potentially cripple an organization's ability to execute its critical business functions.

It impacts everything from insurance calculations to budgeting to electronic data transfer. Without Year 2000 compliance, operations will grind to a halt.

The computer software industry is addressing this challenge by creating software tools that reduce the time, cost and effort to make information systems properly function during the Year 2000 transition. Today, I will describe the challenges inherent in a Year 2000 initiative and how computer software can address these challenges by automating many of the tasks required to make information systems Year 2000 compliant.

Solutions can be considered in four phases. The identification and analysis phase, where entire software application inventories are inspected and potential date processing flaws are identified; the triage phase where the decision is made between correction and replacement of that particular software application; the correction phase where the changes are made and complete testing is performed; and, finally, the implementation phase where applications are reintroduced into production.

At Computer Associates, we have over 500 separate software products which had to be checked for Year 2000 compliance. We used our own software tools for each phase of the effort and found that this automation eliminated considerable human errors, intolerable delays and missed deadlines.

Selecting the right tools is an important step. But, training is equally important.

Organizations that take advantage of vendor-provided start-up services will quickly become more productive. Moreover, software vendors must support the tools with help desk hotlines and comprehensive documentation.

Some organizations will be able to rely on their own staffs, while others will require outside consulting services. A team approach between software vendors and service providers is essential for high quality results in the shortest possible time frame.

The conversion task that awaits us is daunting. But, automated tools make it possible for organizations to complete the conversion before it's too late.

The Year 2000 challenge is formidable. The deadline won't move. And, we have a limited pool of resources.

Ironically, even though this problem stems from software, software is our only salvation.

Thank you.

[The prepared statement of Mr. Sokol follows:]

Testimony of Marc Sokol

Computer Associates International, Inc.

U.S. House of Representatives Committee on Science, Subcommittee on Technology

May 14, 1996

The millennium date change can potentially cripple an organization's ability to execute its critical-business functions. It can potentially impact everything from insurance calculations to budgeting to electronic data transfer. Without Year 2000 compliance, business will grind to a halt.

The software industry has addressed these challenges by developing tools that reduce the time, cost and effort to make MIS systems Year 2000 compliant. This paper describes the challenges inherent in a Year 2000 initiative, and how computer software is addressing these challenges by automating many of the tasks required to make information systems Year 2000 compliant.

More than 80% of Computer Associates applications, for example, are already Year 2000 compliant. CA quickly achieved compliance by using our own tools for each phase of the conversion to help automate the process. Automation eliminated human errors, intolerable delays and missed deadlines.

By successfully analyzing and correcting millions of lines of code, we have gained valuable insight into effective and efficient Year 2000 initiatives. We have extensive experience in conversion software and services, which are typically required for a successful Year 2000 initiative.

Selecting the right tools is an important step. But people who use these tools must also be properly trained. Organizations that take advantage of vendor-provided startup services will quickly become productive. Moreover, software vendors must support the tools with help desk hotlines and comprehensive documentation.

Some organizations will be able to rely on their own staffs; while others will require outside consulting services to convert their applications. When selecting software, organizations that require additional help should look for consultants already familiar with the software used for the compliance effort. A team approach between software vendors and service providers is essential for high-quality results in the shortest possible time frame.

The Dilemma of Legacy Applications

Legacy applications run organizations today – but what is their future? Should these systems be discarded in favor of new applications that take advantage of distributed client/server computing? Or should information systems management continue to devote the majority of its resources to insure that these systems remain in production? One thing is certain: legacy applications cannot be ignored. They embody the critical processing an organization needs to perform and maintain uninterrupted core business computing. For the most part, it is these applications, developed over many years, that IS organizations must enable for Year 2000 processing.

Legacy applications are distinguished by several factors:

- They are normally very large with complex interactions that are not well understood.
- They usually execute on mainframe systems.
- The vast majority are written in COBOL. In fact, studies show that COBOL accounts for almost 80% of all production applications in use today.
- Because of their age, they are written in older COBOL dialects which may not intrinsically support four-digit dates.
- They interact with pre-relational data sources like IMS and VSAM but need to work with relational data as well.
- The original author of the program is often no longer available to explain the reasoning behind the code.
- Source code — the language that describes the program — is often no longer available.

The challenge is to provide Year 2000 support in huge mainframe-based COBOL applications that won't easily compute century dates and that work with non-standard proprietary data formats. These applications — commonly referred to as "spaghetti code" — were written years ago, often before the advent of structured computing techniques. And even if the original programmers are still on staff, they've probably long since forgotten the logic behind some of the more esoteric routines.

Studies consistently show that maintenance of legacy applications is the most resource-intensive task for IS management. And the biggest portion of maintenance is the time spent understanding the existing code. Naturally, this analysis needs to be done before the first line of code or database construct can be changed. Adding support for Year 2000 date processing will necessitate a large maintenance effort, with considerable source code, database, and operating environment analysis. Unfortunately, if this task is not managed correctly, the result may be ballooning costs, unmotivated IS staff, and dissatisfied end users, all of which disrupt an organization's ability to compete.

What is the cost estimate of Year 2000 maintenance? Currently, major consulting organizations are quoting $0.30 to $0.50 per line of code, depending on system complexity and frequency of date references in the code. The complexity of the module increases with the frequency of date reference and can consequently increase the per program modification costs. Overall, it is expected that the typical company (with 10 million lines of code or 5,000 - 10,000 programs to be modified) is facing costs of an estimated $3 million to $5 million for a manual conversion.

How long would it take to complete this conversion? This same company would need 24 man-years to complete the coding effort. And this does not include testing necessary to guarantee that the manual implementation is complete.

Understanding Your Options

Think about it: the typical company needs to invest millions of dollars and several years just to keep its existing systems in production! That means no new enhancements, no graphical user interface applications, no client/server or object oriented implementations, and no new technology of any sort.

So why not simply get rid of these systems and build new distributed systems that exploit advances in computer technology with built-in support for century dates? Why invest the resources needed to adequately understand these legacy systems? Because rewriting these systems entails enormous cost and risk. It has been calculated that if you rewrote one line of COBOL code per minute, it would take approximately 360,000 man years to rewrite all the COBOL code in production today. And the resulting quality assurance effort would be enormous.

Another solution is to evaluate each existing application to determine its degree of technical obsolescence and ability to support changing business requirements.

Most organizations will find that the optimal solution is not global re-engineering, but a balanced integration of revitalization and re-engineering.

A major advantage of this approach is the ability to implement Year 2000 date support without compromising parallel

IS plans for distributed processing or other technology advancements.

Legacy applications can be exploited through reuse of critical components, such as callable date routines that understand century date requirements. In addition, component reuse promotes standardization and consistency, improves quality, and reduces redundant software maintenance efforts.

Once an evaluation is complete, there are several ways to proceed. One approach is to review all affected data structures and determine the code changes and field expansions required to accommodate a full four-digit year representation. A second approach is to modify only the source code and leave the data records intact, assuming the program logic handles the evaluation of a two-digit year value.

The first approach is more comprehensive, but often more expensive. The second may be less expensive, but makes logical assumptions about dates that may not always hold true. For example, if the application projects dates only into the future, it's safe to draw conclusions about the century from the year information (if the year value is *00,* the application processes it as *2000).* However, an application that processes dates of birth cannot safely draw the same conclusion because multiple centuries are applicable.

Both approaches will be expensive, time-consuming, and tedious. Tools designed to automate software maintenance, however, will greatly increase efficiency. Such tools and techniques can often save half the cost of the manual process. The tools will readily pay for themselves by reducing errors and increasing efficiency.

Year 2000 Road Map

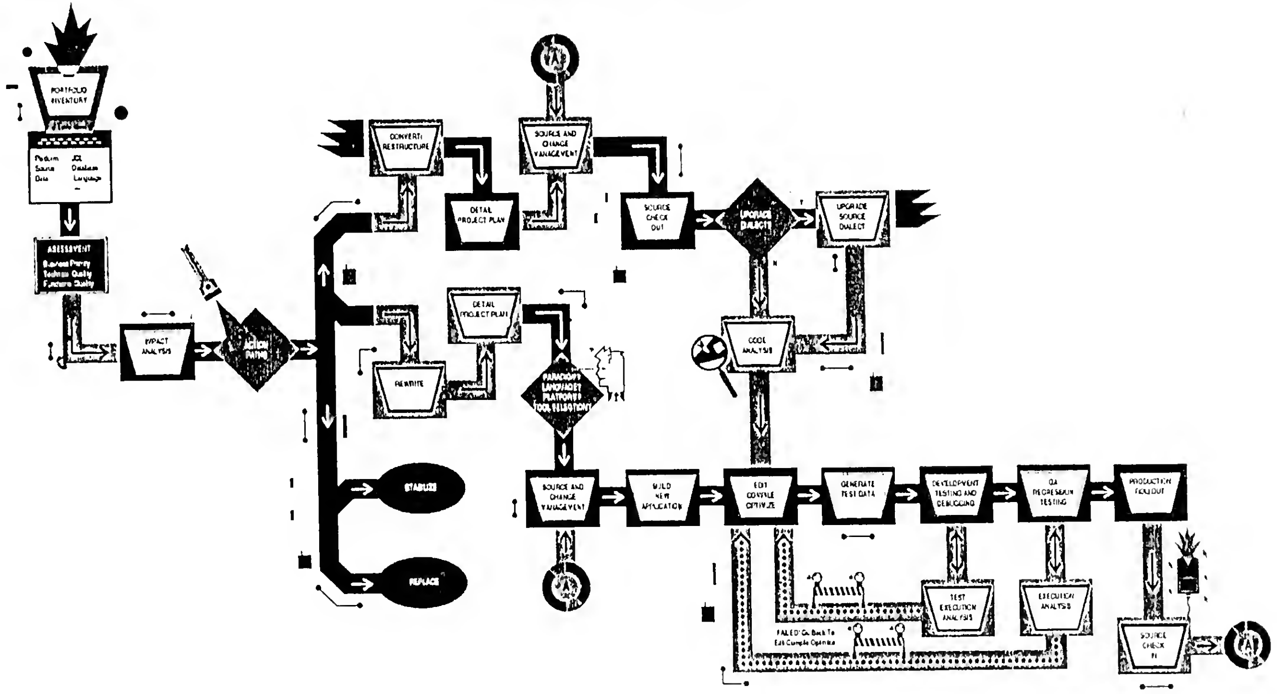
Steps in a Year 2000 Initiative

What are the necessary steps to achieve Year 2000 compliance? To illustrate the process, we have created the "Year 2000 Road Map". It outlines the choices and steps that should be considered when undertaking a Year 2000 initiative. Understanding these steps helps development and maintenance teams make informed decisions in sync with their organizations' goals.

The following sections detail some of the steps that frame the "Year 2000 Road Map":

- Planning
- Inventory
- Impact analysis
- Source code analysis
- Source code conversion
- Edit/compile/link utilities
- Testing and debugging
- Regression testing
- Life cycle management

Year 2000 Road Map


PORTFOLIO INVENTORY
Platform
Source
Data
JCL
Database
Language
ASSESSMENT
Business Priority
Technical Quality
Functional Quality
IMPACT ANALYSIS
ACTION PATHS
CONVERT/ RESTRUCTURE
DETAIL PROJECT PLAN
SOURCE AND CHANGE MANAGEMENT
SOURCE CHECK OUT
UPGRADE DIALECT?
UPGRADE SOURCE DIALECT
CODE ANALYSIS
REWRITE
DETAIL PROJECT PLAN
PARADIGM/ LANGUAGE/ PLATFORM/ TOOL SELECTION
STABILIZE
REPLACE
SOURCE AND CHANGE MANAGEMENT
BUILD NEW APPLICATION
EDIT COMPILE OPTIMIZE
GENERATE TEST DATA
DEVELOPMENT TESTING AND DEBUGGING
QA REGRESSION TESTING
PRODUCTION ROLLOUT
TEST EXECUTION ANALYSIS
EXECUTION ANALYSIS
FAILED? Go Back To Edit Compile Optimize
SOURCE CHECK IN

Charting Your Course Of Action, From Portfolio Inventory To Project Completion.

Source: Computer Associates International, Inc.

Portfolio Assessment

The first step is to conduct a portfolio assessment by obtaining a comprehensive inventory of all system related components, including execution platform, source code, job control, databases, data, and program language. Then, each system or application must be assessed to determine whether:

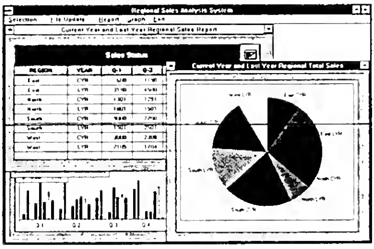
- There is a business need to keep it
- It is technologically capable of being Year 2000-compliant
- The current functionality needs to be restructured for improved use and maintainability.

Having completed the portfolio assessment, you can now take multiple action paths, each one dependent on the assessment outcome:

- Convert or restructure the existing code
- Rewrite the application or system in another language, possibly on another platform
- Take no further action and effectively stabilize the system as no longer active
- Replace the system with a comparable "off-the-shelf" package product.

Application Re-Engineering

Whether the choice is to redesign, rewrite, or merely revitalize systems, new solutions are needed. Multi-platform, object oriented, event-driven systems will support business into the next century. In some instances, new client/server applications should be developed to replace existing legacy applications instead of converting them.


Regional Sales Analysis System
Selection
File Update
Report
Graph
Exit
Current Year and Last Year Regional Sales Report
Sales Status
REGION
YEAR
Q-1
Q-2
Current Year and Last Year Regional Total Sales
Q1
Q2
Q3
Q4

Convert/Restructure

If you decide to take the conversion or restructuring path, you should ensure that all the system components are kept in a source management repository with change control enforced. This is essential to ensure that any and all modifications to the components are monitored and secured.

Impact Analysis

Once safely stored, one or multiple systems can be included in a Year 2000 impact analysis that will provide a detailed set of reports that estimate the scope and magnitude of the modification effort. The report data can be used to build project schedules that identify the manpower and timeline, which may result in a decision not to convert or restructure, but instead to take one of the other action paths.

However, since most systems are mission-critical, conversion or restructuring of the systems will most likely be undertaken.

Source Code Analysis

Once you are committed to a system conversion, determine if the language dialect needs to be updated to support the next millennium. If necessary, the source code must be modified to comply with the new dialect syntax and standards.

To fully comprehend the information provided by the impact analysis reports, individual program logic flow analysis must be performed. This enables the impact report data to be viewed in context with the business logic of the system, thereby allowing the programmer to understand the full implications of any modification.

Conversion and Testing

With the logic analysis complete and the modifications fully understood, the affected source code can be edited, recompiled, and unit tested. However, without the aid of software tools, there is no certainty that the modified code is correct or has successfully executed. Interactive debuggers and execution analysis tools will prove that the modified code is executing and performing as expected.

A test data generator tool helps to expedite the testing process and provide a more rigorous test suite. A generator can create voluminous test data specific to the individual programs and can be tailored to ensure the data forces execution of modified code.

No change can be considered complete until the whole system has passed quality assurance system testing. In this quality assurance environment, the changed programs are executed in concert in an emulated production environment. Here the emphasis is on regression testing to ensure that no new problems have been introduced by the code modification. An automated regression test suite provides the accuracy, speed, and repetition required to ensure the modifications are ready for production use, saving invaluable time and money.

If one or more of the conversion activities is likely to encounter an error, which necessitates that the process be repeated, these savings are considerable. In addition, the availability of workstation-based facilities that allow the use of GUI tools can also improve the conversion and testing effort by displaying information in a graphical, easy-to-comprehend manner.

Rewrite

If your initial decision was to rewrite the system, the activities differ slightly, but encompass many of the same steps used in a conversion project. Many new questions, however, must be answered:

- What does the existing system provide?
- What platform will the application execute on?
- What language will the system be written in?
- What paradigm will it embrace?

Once these decisions have been made, the code should be written and secured in a change control source management repository.

The need to then edit, compile, test, and perform quality assurance exists just as for converted systems.

Tool Selection

The selection of tools to improve productivity during application development or modification defines the infrastructure not only for Year 2000 projects, but for all future application development. It forms the framework for a new development methodology that will improve application maintainability and reusability through better understanding of the business logic that has been systematically built within the enterprise over many years.

Change and Configuration Management

When undertaking a maintenance project on the scale of the typical organization's Year 2000 changes, change and configuration control procedures must be in place, both to protect the new investment in applications, as well as to ensure a smooth implementation. These concepts are well accepted in most IS organizations. However, because many companies lacked proper procedures or enforcement of these concepts in the past, they may discover that they are running load modules that can no longer be easily recreated. Once the source is found or recreated, organizations will need products that control the turnover process and ensure not only that implementation is a controlled process, but also that source and load modules remain in sync. Typically, this involves using source management libraries and supporting software services.

Impact Analysis

Impact analysis is the process of analyzing your portfolio to assess the scope and cost of converting your applications. First, find every date field in every program and copybook, and decide whether to leave it alone or modify it to accommodate century information.

Finding date fields isn't easy; seldom are they well documented or defined with consistent names. Tools that can intelligently analyze source code are essential. This task is far beyond the capabilities of simple string-based tools that scan files looking for string patterns. What is needed are tools that understand the syntax and grammar of data processing languages, especially COBOL. To be useful, these tools need to understand:

- The difference between a comment and a source statement
- Redefinitions of data
- Data usages, such as packed versus binary
- The context of date fields: inside a file definition, Working-Storage or Linkage Section

Once all date fields have been located, several reports are needed. The CIO needs to know the cost of converting the enterprise; the development manager needs to know the cost of converting each application; the programmer responsible for converting and testing programs needs to know every line that contains a reference to a date field.

To accurately estimate the cost of converting a program, factors such as line of code costs need to be assigned by the user. Also, program attributes such as the number of files and the complexity of the code affect the estimated conversion costs. Surprisingly, the number of files read or written by a program contribute to the conversion costs even for files that contain no date fields! The effort required to test a program increases considerably when it uses multiple files because test data extracts are normally required for each file in order to test the program. Test data generation, as I will explain later, is an area that benefits from automation.

Factors to consider when evaluating an impact assessment tool include:

- **Multi-Platform Support**: The flexibility to perform impact analysis wherever the source resides is critically important. The ability to off-load some analysis work to the PC workstation can dramatically improve productivity.
- **System Date Usage:** The detection of statements that retrieve the date from the operating system is essential. This varies by language and dialect. CICS programs, for example, retrieve the date quite differently from batch programs.
- **Pattern List:** To help locate fields that potentially hold dates, a pattern recognition algorithm should identify candidate data items based upon patterns supplied by the user.
- **Nested and Non-Standard Copybooks** Programs can have nested copybooks and can use copybooks stored in non-standard library formats. Typically, these copybooks are expanded into a program at compilation time by a preprocessor. For maximum usefulness, the impact analysis tool should be able to access a wide variety of library formats and recognize a variety of non-standard "COPY" statements.
- **Source Access:** Another very real concern is access to source where it lives; this is, can you analyze your source in its native source library or do you have to first stage it in another medium? If so, what are the hidden costs involved in terms of additional DASD (Direct Access Storage Device) and import, export and download procedures?

Analysis should provide statistics for the correction phases of the project. Again, flexibility in platform choice is a requirement. Information useful in a conversion effort includes:

- Number of programs analyzed
- List of programs that contain date references and the lines affected along with associated programs, files, and copybooks
- Reports that cross-reference CALLs and COPYs, which are essential in order to understand dependencies

Source Code Logic Analysis and Compilation

All applications will need to be examined, and in most cases, modified, to eliminate calculations based on two-digit dates. Performing the vast number of recompiles and testing scenarios on the mainframe would be overwhelming. One needs to be able to take advantage of less expensive desktop processing power.

Off-loading analysis, compilation and testing to workstations has numerous advantages over using the mainframe:

- Reduced cost with no charge-back
- Faster turnaround
- A more intuitive interface
- Stronger and more powerful visualization capabilities via graphics and color

PC-based compilers and CICS emulators must support older and current versions of the mainframe systems. Implementation of integrated mainframe/PC life cycle management solutions is essential. This will tie mainframes to PCs in cost effective ways and protect the integrity of the source.

Source Code Dialect Migration

Applications in older COBOL dialects such as OS/VS COBOL that are not candidates for replacement or conversion pose a special problem. It is vital to ensure that they can function and be maintained into the 21st century. Manual conversion to a current COBOL dialect is tedious, error prone, and time-consuming. Automated conversion is a requirement.

Testing and Debugging

A maintenance project with the scope and impact of the typical Year 2000 project requires improved efficiencies in the testing and debugging phases of the project. Thorough testing is essential to ensure that the conversion is accurate and complete. This is best accomplished through workstation-based tools. Organizations that choose not to implement such tools must test on the mainframe. Even those organizations that take advantage of the workstation environment must perform final testing on the mainframe.

When test errors occur, the ability to quickly debug the application is essential. Make sure that your Year 2000 plan includes debuggers for both batch and on-line applications.

Test Data Generation

Creating test data to exercise a modified application can also be resource intensive and time-consuming. The use of accurate and complete test data ensures that the altered code is executed under all possible circumstances. Manual data creation can be eliminated by using a test data generator tool. This significantly reduces the time and effort needed to create test files, while producing valid, accurate, and customized data.

With a data generation tool, test data can be recreated automatically. Dynamically generating the data saves valuable DASD.

Regression Testing

Testing requirements are the same, regardless of where development takes place. Automating testing for on-line applications reduces the workload and allows more complete testing. By using tools that automatically record and re-execute test scripts, consistent and complete testing of applications with a minimum of manual intervention is ensured.

In addition, since the functionality of the application is unlikely to have changes, these test scripts can be captured from existing production application data.

Summary

CA's efforts, both with our own software systems and those of our clients, have given us the real-life experience to formulate the road map for reference and guidance in a Year 2000 endeavor.

The conversion task that awaits all MIS departments is daunting, but automated tools make it possible for organizations to complete this conversion before it's too late and to reduce the conversion cost and schedule. Equally important, automation limits human error and helps ensure that the converted applications perform properly in the next century. The Year 2000 challenge is formidable, but fully tested software solutions are available for each step of the way.

© 1996 Computer Associates International, Inc., One Computer Associates Plaza, Islandia, N.Y. 11788-7000.

All product names referenced herein are trademarks of their respective companies.

Chairwoman MORELLA. Thank you very much. And, I thank all of the panel. I wanted to, first of all, direct a couple of questions to Dr. Hebner, particularly because of what we heard from the first panel and, you know, even what has been stated through this panel, too.

I wondered why NIST has not changed the standard for year fields in federal computer programs. In fact, even now the NIST Federal Information Processing Standard that was submitted March 25th of this year only recommends that four digit year elements be used by federal agencies for purposes of electronic data interchange.

I just wonder why stronger language wasn't used.

Mr. HEBNER. We have looked at the situation and we think that there is not a—we don't anticipate a problem with noncompliance. We expect that the agencies will choose to make a change which will make them consistent with everyone else.

There may possibly be some applications, that we are not aware of, in which that change is not the appropriate change. And, we did not want to lock people into not being able to make the intelligent decision with their own resources.

But, we strongly recommend that they adopt the Federal Information Processing Standard, which is based on and consistent with the voluntary standard that was promulgated by ANSI, the American National Standards Institute.

Chairwoman MORELLA. What if they don't?

Mr. HEBNER. Well, their computer systems crash if they—if everyone else does it and they don't. So, they have the—there is an incentive for them to be consistent with everything else, everyone else.

The purpose of the standard is to tell them how to do that.

Chairwoman MORELLA. But, we didn't hear—we have not heard that this problem is even being taken that seriously with the exception of some companies here represented and an agency. And, it seems to be the feeling that if we have required standards that we—I understand what you are saying about making it an absolute requirement, but by pushing, you know, businesses and federal agencies and other agencies to look to, that would make a difference.

Mr. HEBNER. Well, I think that is what we are doing.

Chairwoman MORELLA. Maybe requiring. Maybe requiring is the way to do it.

Mr. HEBNER. Maybe it is. We don't—NIST does not have a regulatory function, and we tend not to require.

I mean, in principle, we could if we were directed to. But, we have not found—it is not our analysis—that there is going to be a problem with noncompliance.

And, I mean, this is—obviously, we do not know that. We are going to monitor the situation and we are cooperating with the federal group to see if compliance—noncompliance is a problem. But, we do not anticipate noncompliance.

We expect that if someone will say, "This is what we should do," and everyone is in the process of making their changes, they will make the change to the standard that everyone else is using.

Chairwoman MORELLA. Well, 65 percent of businesses, according to Mr. de Jager, have just ignored it, haven't even started on that path. Your feeling is that they will suddenly catch on?

Mr. HEBNER. Well, I think you are asking me a different question.

Chairwoman MORELLA. Yes, I know. I am switching——

Mr. HEBNER. I mean, you switched.

Chairwoman MORELLA. (continuing) —to private, right.

Mr. HEBNER. Well, not only just to private, but the standard is for those who want to change. The separate issue is how do you——

Chairwoman MORELLA. Educate people.

Mr. HEBNER. (continuing) —educate people if they want to change. The standard is set up under the assumption that people have become educated, they want to change and they know to what. And, then this recommends to them to what they want to make the change.

There is a separate issue, which is not a standards issue, which is how effectively are we getting out the word to everyone. Is there someone out there who does not realize they have the problem?

And, that's what this hearing is about, to look at those issues. But, that is not a standards issue.

Chairwoman MORELLA. Okay. Do you see a stronger role in NIST in trying to educate and establish or make sure that there is movement toward that standard?

Do you see that NIST could play a stronger role in this?

Mr. HEBNER. Well, I think we are trying to play the most appropriate role for NIST. And, that is, we are promulgating the standard and we are making it obvious in every way that we can through all of our normal distribution channels as to what the standard is.

Again, I don't think the issue is that people are going to be confused about what the standard is. I think the issue is when management of various organizations make the commitment that they want to make the change.

Chairwoman MORELLA. Okay.

Mr. HEBNER. Which is a larger issue, but it is not a NIST issue.

Chairwoman MORELLA. But, you are saying that you are not—this is not part of your jurisdiction——

Mr. HEBNER. Right.

Chairwoman MORELLA. (continuing) —or a requirement for you to do that.

Mr. HEBNER. Right.

Chairwoman MORELLA. Turning to some of our other panelists, it appears to me that you are all saying that there are solutions that are available. And, I know that, Ms. McDuffie, you seem pretty confident that IBM is making them all compatible and that individuals as well as businesses can adapt.

Are you finding that other companies are doing that? How do you see the entire picture, going beyond IBM?

Ms. MCDUFFIE. Let me use some examples. I mentioned to you part of our business is working with customers who ask us for some help.

And, we announced some assistance last fall. And, some of the feedback from some of those engagements would give you some of this, I think, examples of what you are asking.

What we are saying is those companies are treating this as a business issue. It's how they are going to run the business on Day One of Year 2000.

And, many of them are in re-engineering projects that says how do they make themselves more competitive and, therefore, change the IT systems to support those business issues. Let me give you a couple of examples.

We are working with one medium sized utility. It's a subsidiary. And, the engagement asked us to do the complete process.

We did an assessment. We actually implemented the coding changes. We did the testing and now working with the customer to put the units of work—it's two units of work, think of that as two application suites, like billing and order entry or something, put those back into production.

It has taken us about eight months. And, the customer projects that they will spend about $1 million. That's a medium sized—a subsidiary of a medium sized company.

Another engagement that we have is a large insurance company. We got about 75 consulting resources engaged.

This one, we are just finishing the assessment. They estimate that it will be about a $20 million project.

But, they have a plan in place. And, I think the key here is the plan, is that the first—first, it's awareness and then it's management commitment to a funding prioritization in order to get a plan complete.

This particular company plans to have their key applications back in production by January 1, 1998 so that they have time, you know, to actually exercise them before hitting Year 2000.

And, so what we are seeing is that the companies who view this as a business objective and how they will run their business in Year 2000 are stepping up to the plate. They are either setting aside a project where they do the conversion themselves or they are hiring outside resources.

And, as Congressman Calvert asked earlier, there are lots of companies going into business to help support customers with that move.

Chairwoman MORELLA. And, Mr. Ingram, also to you, what if some of your clients or customers do not do that?

What would happen if it's just one sided?

Mr. INGRAM. Let me respond a little bit to the prior question also.

Chairwoman MORELLA. Yes.

Mr. INGRAM. I think it is a problem. I think it's a severe problem.

When resources are short, it's a problem. When we don't have enough people to work on this or we don't have the dollars set aside, it is a problem. When it costs us at least around 50 cents per line of code to convert code and then maybe another 50 cents to 60 cents to test it and to get it implemented, I think that's a problem.

If we've got current system plans to implement things today that we need in our businesses or that we need in our government—and

I will make the assumption that those systems are important or we wouldn't be implementing them in the first place—then I don't feel we have the money set aside to tackle the Year 2000 issues. So, if we are going to throw away our current systems, then we've got the money.

But, I don't think that is one of our options either. So, I think we have a real problem.

And, it's a resource problem. And, then that goes back to my other statement that if we've got resource problems then we have got to effectively manage those resource problems.

Chairwoman MORELLA. I wondered, I guess, back to the other question. How are companies that serve as your customers or suppliers handling the conversion?

And, what if they don't? I mean, how do they interact?

Mr. INGRAM. Well, a lot of the companies that serve as our suppliers are already tackling the issue. They are making the changes in their software, like Computer Associates or like IBM. They are tackling those problems.

The ones that don't tackle those problems are going to have problems in the marketplace. They are going to have severe problems in the marketplace, because their systems won't talk.

I would also like to commend NIST for putting out a standard. Finally, somebody has put out a standard.

We have existed for years without one. And, if we don't adopt a standard of some sort for data interchange, this folly is going to go on. It's going to go on past the Year 2000.

We will solve the Year 2000 someway, through brute force or conversions or whatever. So, systems will run, because we can't afford not to have them run somehow.

But, after 2000, if we don't adopt a standard for dates, we will continue to have data interchange problems. And, it won't be that systems won't run. They just won't talk to each other.

Chairwoman MORELLA. Should we require more from NIST?

Mr. INGRAM. I think we ought to have a set standard, a required standard, for data interchange.

Now, there will probably have to be a waiver, a process of some sort, because I am not sure that all of the standards that are already out there for other types of data interchange like EDI and EC already don't have a standard in them that would have to be converted or changed or, again, interfaced between. So, there might be a requirement to have some sort of a waiver process that would have to go through some sort of checking capability.

Chairwoman MORELLA. Mr. Sokol, have you decided that there might be a role for this Congress to fill in making sure that we move forward to correct the Year 2000 problem?

Mr. SOKOL. Well, much like Mr. de Jager, I think that effectively putting out the message that all the companies and organizations out there should go up into their attic and look at all of their musty boxes of source code and see exactly what they have is the most important. Just doing an assessment of what's involved, it may be better than you think or it may be worse than you think. But, at least you will be able to put your hands around it.

And, I think that impact analysis of understanding of what you have to do—there are many, many computer programs out there that may not need change. And, then there are many computer programs out there that not only do they need a change but, as was said before, you may not even have the source code in which to change them.

We talk about the software industry being pretty responsive in updating their own programs. But, there are many companies and organizations out there that are running software from computer software companies that don't exist anymore. And, there is no one there to fix those programs.

Or programs that were written 25 years ago by employees that are not in that organization anymore or x-amount of products that are being run that really there is no way of fixing them. And, a replacement strategy is required.

So, I think if there was one thing that Congress could do would be to get the word out so that businesses and organizations of all sizes would immediately assess the impact of the Year 2000 transition on their own software inventory.

Chairwoman MORELLA. Interesting. Mr. Tanner.

Mr. TANNER. Thank you all for your testimony. Do any of you, individually or collectively, believe that voluntary standards and the compliance that we hope we get with those is sufficient?

Or, do we need to consider some sort of public policy in a statute to bring this to the public's attention?

Mr. SOKOL. I will just go first. I think that actually the effect of going for public policy on a standard, the outgrowth of the press coverage, will probably have more of a benefit on people looking at their problem than anything else.

You know, decreeing——

Mr. TANNER. Just a statute on the exchange of information and a standard for that.

Mr. SOKOL. I understand that. And, I think that's valuable.

But, as I said, the challenge is just having people, you know, admit to themselves that there is a problem within their own organization. Many, many organizations are several steps away from even worrying about converting their programs to fit any sort of standard.

So, I think the standard is important. And, beyond just the interchange of information.

Mr. TANNER. Barbara?

Ms. MCDUFFIE. Yes, Congressman Tanner. I thought about this a lot, as I was invited to participate here.

My belief is that if you make people aware, there will be—and encourage them to act on their own self-interest that there will be enough self-interest to get them to move forward. And, if we think there should be standards, I would like to see that done by the industry where we can have, you know, a collective set of folks who work in the industry be able to participate fully in being able to set that standard.

What we have done in our planning and implementation guide is offer some guidance, if you will. But, each company, I think, has the opportunity here, because the systems are so complex that mandating a standard may make it more complicated than if you let the companies make the changes, because the technical solution should be very simple.

The complexity here is the amount, the volume—like, in our mainframe class of programs, our customers who have that, there's over a trillion lines of application code. All of that needs to be assessed to determine which ones are date sensitive.

So, it's not so much a standard that needs to be done as just taking an assessment and then taking action.

Mr. TANNER. Dr. Hebner.

Mr. HEBNER. I will endorse what I've heard. As far as we can determine, the issue is not that the—the problem is not the fact that the standard is voluntary rather than mandatory. The problem is that people don't understand the magnitude of their problem yet.

And, I think that once this understanding is in place, we have confidence that a voluntary standard will be——

Mr. TANNER. Well, let me ask you a different question, then. Do you think we—how do we communicate the problem so people can understand it?

Do you have a clue on that?

Mr. HEBNER. Well, my first answer is that I worry about measurement of the standards and not public relations.

Mr. TANNER. All right.

Mr. HEBNER. But, I think that we are doing—I think that this hearing is a good step in that direction. I think that the companies you have heard from are using every medium available to them to get this information out.

I think the federal government, through its coordinating committee, is trying its best to get the information out. So, I think there is—maybe it has started too late, but I think now there is an intensive effort in place to get the information out to anybody who is interested.

Mr. TANNER. Well, all of those people, I assume, who are doing what they need to do now, if we were to try to pursue enacting some sort of statute about the exchange of data wouldn't be adversely affected. They are already doing it.

The only thing the statute would do would be to send the message to those who otherwise are not clued into what I heard, I thought, would be a cataclysmic event, that they need to do it. In that light—and I'm not—I don't run around writing bills all the time, but if this is as serious as we have been led to believe, then I wonder sometimes whether or not Congress would not be a little derelict in what we are doing if we know that this problem exists and just by somebody in Congress pronouncing to the world they ought to reassess all their programs is not going to do it.You know that as well as I do.

And, I mean, I don't know. I am asking. You all are the experts.

And, I just see that a company going down on its own unfortunately will not be in a vacuum. This utility company that you talked about, you said they are the only ones that are going to get hurt if they don't do what they need to do between now and the Year 2000.

You and I both know that's not true. There's going to be a lot of people who are customers of that utility and others who are going to be adversely affected.

And, so I just—if there was some way to communicate to all the people in the country that need to take action, what I have heard as the almost crisis mode we are in now in 1996, I would be perfectly willing to try it. I don't know that I heard that.

Maybe ya'll can help me with this one.

Mr. INGRAM. Congressman, I would like to reply to that. I think we need to be proactive on this whole issue.

If we don't set the standard, then who is going to set it? Are we going to be reactionary?

Is Europe going to come in and set the standard and then we try to follow that?

And, if we don't do it now, then when are we going to do it?

And, then——

Mr. TANNER. We are going to have to before the Year 2000.

Mr. INGRAM. Well, the Year 2000 to do it is too late. We have already passed the crisis at that point.

So, if we don't do it now, then we may do it too late. We've got the mechanisms out there today, I think, to do some of this.

Current requests for proposals that are out or requests for proposals that will be coming out from the government can put requirements in them. We can have those kinds of things.

Current authorization bills can mention the subject or have the requirements in there. So, I think we've got the means out there to make some of this activity happen.

Mr. TANNER. But, do you agree with Ms. McDuffie who said she thought perhaps a statement of statute about exchange of information would be maybe detrimental rather than helpful?

Mr. INGRAM. No, I don't—I do disagree.

Mr. TANNER. I mean, I don't want to complicate anything. It seems from what I've heard it is complicated enough as it is.

But, if we could be of help—I just see Congress as a public policy group. And, if this is as serious as I think I've been told, then I'm not sure that not taking action would not be a dereliction of duty in this case.

That's all I have to say.

Chairwoman MORELLA. I would like to recognize Mr. Calvert.

Mr. CALVERT. It seems that what we are hearing today is we have a communications problem and we have to communicate this to business and governmental agencies. But, I find that self-interest is the strongest force for change in this country and in this world.

And, survival is a powerful inducement to create the change that is necessary and to get this conversion on track. And, I'm one that is reluctant to pass a law or a new regulation unless it's absolutely necessary.

But, have we done everything to communicate? That's the one subject I would like to get back to.

The software companies themselves, Microsoft, the other software companies, I would suspect that their sales force, which is significant worldwide, dealing with all of the—virtually all of the computers out there, are working to communicate that we need to convert. And, businesses out there, both small and large, if they intend to survive and to do business, must convert.

And, as this time becomes shorter, are we getting better reactions from these companies?

Mr. SOKOL. We have about 3,000 worldwide sales people that sell our products. And, the way that we do the communications is the top/down method.

We meet very frequently with the chief information officers of large government organizations and businesses around the world. And, from that level down, discuss the problem, discuss potential solutions.

And, at that level, certainly—let's say, the top 2,000 businesses in the world, there's a lot of concern. And, in one way or the other, those businesses have started some sort of plan.

My belief of what additional communications can occur would be more widespread, sort of a groundswell type of communications that would allow companies that didn't fit into this, let's say, the global 2000, also recognize that the impact can be as devastating on them as it can be on a $50 billion manufacturer.

Ms. MCDUFFIE. Congressman Calvert, I would like to respond to that. We have 20,000 salesmen worldwide working with my partners here as well.

What we've done this year, as part of an integrated marketing program, is arm our salesmen with the presentation for awareness. We also then armed them with what I just talked to you about earlier, the four phases that we have announced to help them move.

As such, we are seeing more and more people step up to the plate and take action. Now, specifically what I would like to see Congress do is I would like to see you recommend and encourage that the agencies each have a plan, to have a plan for completion by a specific date and that you show your support in helping them with priority funding.

Now, in the case of Social Security, I heard him say he didn't need any right now. I don't know if that's the case across the board.

What I would then like to see you do is take an example and publicize it to the broadest audience possible as to here is what we are doing in the U.S. government.

Mr. CALVERT. That's a great opportunity. Survival/self-interest creates opportunity.

And, we talked about startup businesses that are getting going right now. I suspect that there are a lot of people who will get into this very shortly.

One other question. You mentioned IBM 370s. How many of those do you still have out there?

Ms. MCDUFFIE. There are still quite a few out there. What we are doing this year is we have also announced—we have totally revitalized the technology to turn the mainframe into a large scale server.

And, I personally don't know the exact number. I can get that and get back to you.

Mr. CALVERT. Fine. The reason I asked that question, I suspect the United States government has a significant amount of 370s still on line.

And, that type of hardware that cannot be converted, you mentioned earlier that we cannot make the changes within that hardware to make it work. So, all of that hardware will have to be scrapped.

Ms. McDUFFIE. That's one way of looking at it. Let me just share with you, though, what a lot of the customers who are going from 370 to our newer technology—the newer technology, we follow the cost curve, now inside those big processors are the same chips that are used in PCs, so they are much cheaper. And, so what customers are finding is that they can replace those older systems and pay for them within a year just on the savings on electricity, on space and on cooling.

Mr. CALVERT. You know, a lot of businesses look at quarters rather than years. So——

Ms. McDUFFIE. Okay.

Mr. CALVERT. (continuing) —I don't know if we are any different. Thank you, Madam Chairman.

Chairwoman MORELLA. Mr. Gutknecht.

Mr. GUTKNECHT. I just wanted to follow up with what Ms. McDuffie was saying. And, that is, I am surprised that there are that many businesses that still have this really old technology out there when they can be buying all this new IBM stuff that is manufactured in Rochester, Minnesota.

[Laughter.]

Ms. McDUFFIE. Thank you.

Mr. GUTKNECHT. And, I think, frankly, this first hearing is a good start. I was going to ask Dr. Hebner whether or not—maybe one of the answers is—and this is just an idea, because this is both a national and international problem, if you will, would it be appropriate for NIST to sponsor a national or international symposium just to get more information about this whole thing out there?

And, I guess, the other question that is sort of coming up here is the Social Security Administration is sort of taking the lead on that. And, I guess the Committee, if I hear what the Subcommittee is saying is maybe they are not the—I mean, nothing against Social Security, but perhaps it is NIST that ought to be taking more the lead on coordinating the activities of this.

Can you comment on that?

Mr. HEBNER. Yes, I will comment on that. Not only am I not being in a non-acquisitive mode, but I think it's very appropriate that NIST not take the lead in this because, as you have heard, this is an issue where it's organizations that own legacy software that have date sensitive programs, that have the real problem.

Our job is to worry about the measurements and the standards. That is, in our view, a very small part of the problem.

The problem is more organizational, gathering of resources, both financial and management resources and technical resources, to solve the problem, to scope the problem, to coordinate with the people who are giving you data and the people with whom you are giving data. There are standards there for how to do that.

But, to organize the human endeavor, to get that done, the decision was made that that is better done through the Coordinating Committee. And, I have no reason to second-guess that right now.

We are very happy to participate in that committee. If there is a need for us to have an international symposium on the technology or, even more broadly, on any aspect of the program, we will be happy to do that.

Mr. GUTKNECHT. Thank you, Madam Chair.

Chairwoman MORELLA. I would like to recognize Ms. Cubin for any questions she may have.

Ms. CUBIN. Thank you, Madam Chairman. I don't have any questions at this time.

Chairwoman MORELLA. Dr. Hebner, I think that OMB has asked the agencies to come up with an assessment.

Mr. HEBNER. Yes.

Chairwoman MORELLA. I don't know whether they added a date certain on that or not, because I guess it was like a month ago that this went out. Was there a date on that?

Mr. HEBNER. I don't know. I don't know if there was a date on that.

Chairwoman MORELLA. I mean, that's the way to proceed, then, is——

Mr. HEBNER. Yes.

Chairwoman MORELLA. (continuing) —to make sure that they do respond so that we can then pick up on what Mr. Sokol said on—at least in terms of the federal government.

I guess also what you are saying is that one of our problems is the idea of education. People do not realize the urgency.

They don't—and maybe they don't know how to handle it either. Maybe they don't see it quite as simply as you do, Ms. McDuffie or Mr. Sokol or Mr. Ingram.

I wonder, why can't software solve the problem? Why can't you develop, Mr. Sokol, that so-called silver bullet that would handle things?

Mr. SOKOL. Actually, I was going to come back. I will answer that by way of talking to Congressman Gutknecht's question about old technology.

Back in the 1970s, I was probably one of many people that contributed greatly to the problem that we have today. And, what's interesting is that back then when we were writing the software and saying to ourselves, "Well, two-digits, you know, what about the Year 2000," generally the answer was, "They will have replaced this many times by the Year 2000."

Now, I remember earlier someone saying that the computer industry tends to be optimistic about a lot of things and deadlines being one of them. But, we certainly believed that the accounting systems that we wrote in 1972 and 1973 would no way be in production today. And, I think they may be sort of the mid-age of some of the software running today, because they are stuff from the 1960s.

One of the challenges that you have, if you look for a silver bullet, is the number of languages out there. There are 2,000 or so languages. Probably 50 of them, at the very least, represent 80 percent of the programs out there.

But, the problem is not just simply looking for something that said, you know, "Date," and looking at what year it is and changing it to four digits. There are all sorts of business rules that are totally enmeshed in the way that dates are stored.

People that say, "If the year is 99, do something. If the year is 27, do something." We spoke earlier about certain systems that have one digit years that had to be changed.

The problem is that someone said you can use software to solve 80 percent of the problem. And, that's not solving 80 percent of all the programs out there. It's about 80 percent of each program out there.

So, it helps a lot. But, really, you know, modern software technology and the algorithms or the rules by which we build that just does not allow you to deal with all the different ways that dates affect programs.

Believe me, if there was a silver bullet—you have maybe 100 companies now and probably 500 companies by the Year 2000 that are searching for it. And, no one has really come up with it.

So, we can help it. We can speed things up. We can assess it quickly. We can change things that are relatively simple to change.

But, you know, you have programs that were written in the 60s and the lowest level language, which was called assembly language, that have no documentation. The programmers have long moved on.

And, you will have a very, very difficult time understanding what those programs do, let alone finding all the places for which dates are a problem.

Chairwoman MORELLA. You don't see some companies, then, making a fortune because they come up with some of the solutions?

Mr. SOKOL. Companies will do very well selling solutions. They will not be 100 percent solutions, that are software talk analogy only.

I mean, with services or consulting, you can solve the problem completely, but there is no—what I like to say, total computer solution to this. It's always either human-assisted computers or computer-assisted humans that are going to solve the problem.

Chairwoman MORELLA. Mr. Tanner.

Mr. TANNER. I find this discussion fascinating, in that we are focusing basically on how to fix the problems that people who want to fix the problem are aware of. What I am concerned about are those thousands unknown to us that don't even know they have a problem.

And, that, to me, I come back to. How do you do that unless the Congress does something to make public policy by way of a statute saying to everybody, "You need to go to a four digit number and this is what the standard is going to be in terms of interfacing with other people?"

Is there a way other than that? I would be glad to entertain it.

Mr. SOKOL. I think it's akin to a safe computing policy that would have to be spread as widely as any other public policy.

Chairwoman MORELLA. Any questions, Mr. Calvert?

Mr. CALVERT. Again, I get back to the communication part of it. I mean, we could pass a law but that doesn't necessarily mean that people are still going to get the message.

And, all the organizations, business organizations, small business organizations, need to participate and get that out.

Just one last comment. If nobody has talked to the Internal Revenue Service yet about converting——

[Laughter.]

Mr. CALVERT. (continuing) —then let's keep them in the dark.

[Laughter.]

Chairwoman MORELLA. Mr. Gutknecht?

Mr. GUTKNECHT. No questions.

Chairwoman MORELLA. Ms. Cubin?

Ms. CUBIN. No questions.

Chairwoman MORELLA. Just one final question. What do you think we should do—first of all, I think that the panel agrees that we should follow through with the OMB requirement that agencies give us an assessment or submit an assessment to OMB by a certain date. We follow through with that.

The assessment comes in and maybe we can then require in some way, however you want to word it, require that they convert.

Private sector, you are saying that we've got to let them know how dire the problem is, educate them to the fact that there is a problem and that there are solutions.

Would you like to add anything about what else you would like us to take away from this hearing where you have invested your time in telling us what you are doing and being here with us? What would be your final words to us?

Mr. Ingram?

Mr. INGRAM. Madam Chairwoman, you addressed this earlier when we first started that we might want to talk about the Year 2000 challenge. And, I think this challenge offers us maybe another set of opportunities that aren't being looked at.

Granted, we have got to solve the Year 2000 problem or programs don't run anymore or they don't run right or they don't communicate. But, we've got an opportunity here to do some things in government and in business that this opens up for us.

We've got a chance to look at our businesses and see what's really going on. We've got a chance to look at our business systems out there and say, "Should we modify them? Should we clean them up?"

We have the opportunity to go through all of our source code and inventory all of that. We've got an opportunity to maybe upgrade our systems.

So, I think in all of this that there is a plum in here, too. So, we've got an opportunity facing us.

Thank you.

Chairwoman MORELLA. If people are so motivated, yes, and companies.

Ms. McDuffie, any final comments?

Ms. MCDUFFIE. Let me just reiterate. I would like to see the agencies encouraged to have a plan and then have you support helping them get the funding for that plan.

I would also like the industry—if standards need to be set, then I would like that to be taken up in ANSI or one of the industry standard bodies so that it is properly reviewed with both sectors, if you will, okay, and not just a public mandate.

And, finally, wherever possible, on the awareness—and you reiterated that, so I don't think I need to talk about that anymore. But, to me, that's our biggest challenge, is reaching the broadest audience, to be able to allow those folks who don't realize it to know they have a problem.

Mr. SOKOL. I think the communications issue—I will go back to again as being the most important. And, I think that we have a tendency, all of us involved in this initiative, to cause the people we talk to to have sort of the chicken little problem and go and feel as though there is no solution—oh, I have X-million lines of code, I have no way of fixing it.

I think what has to be said is that, as Mr. Ingram said, there is an opportunity here but that you should take the opportunity now to assess. And, I think that the communications at the highest level is important.

I think that raising the importance of this issue through Congress is a key foundation of that. But, in general, the population at large, anyone who runs an organization or runs a business, has to realize that this is a path that they have to go through between now and the Year 2000 so that their business can continue to function.

Chairwoman MORELLA. I wonder if even academia has responded to this? You know, when you think about supercomputers and all. I'm not sure.

Well, you've given us a lot of information and firsthand experience as well as your concepts of what we should do. I think ANSI, but maybe even other partnerships, other consortia, whatever, should also begin to have this on their agenda, too.

I want to, on behalf of the Subcommittee, thank you for being with us and the expert testimony and responding to questions. We do leave the record open and hope that you will be able to respond if there are any other questions we would like to get to you in a timely fashion.

Thank you, Dr. Hebner, Mr. Ingram, Ms. McDuffie and Mr. Sokol.

[Whereupon, the hearing was adjourned at 2:52 p.m., Tuesday, May 14, 1996.]

[The following material was received for the record:]

TESTIMONY SUBMITTED BY WILLIAM ULRICH TO THE HOUSE SCIENCE COMMITTEE, SUBCOMMITTEE ON TECHNOLOGY

MAY 14, 1996

It is far too late for organizations to use trial-and-error approaches in defining a process to address the year 2000. Clear-headed planning and the use of a formal approach facilitates the creation of cost effective and readily implementable solutions. Making up on-the-fly solutions is too time consuming and risky for a project with such an unforgiving deadline.

A formal process to address the Year 2000 problem must include project planning templates, planning and implementation guidelines, estimating models, metric definitions, forms, commercial tool usage guidelines and a vendor tools data base. In a Year 2000 project, these components could save thousands of hours in research, planning and implementation effort.

Our project team at the IRS is using TSRM (The Systems Redevelopment Methodology) as the blueprint to guide Year 2000 planning efforts. This gives us the opportunity to make sure that all of the players are working from a similar mindset and approach.

A methodology, sold as a commercial off the shelf (COTS) product as opposed to a consultant's handbook, is more robust, more complete and more useful for year 2000 projects. Off the shelf methodologies are packaged in a way that facilitates quick access to information.

One bottleneck for the U.S. federal government is that methodology procurement teams will attempt to stand in the way of Year 2000 coordinators looking to standardize a common process. If these methodologists are allowed input to this process, the Year 2000 will have come and gone before a decision is made.

The federal government would do well to take a proactive stance on this topic and mandate usage of a common process within U.S. agencies. It would also do a tremendous service to the private sector by encouraging use of a standard process there as well.

Inventing a Year 2000 process, within such a limited timeframe, makes very little sense. Jump-starting a project using a well defined set of guidelines is a much more effective way of utilizing the skills of in-house personnel and consultants.

William Ulrich is president and founder of California-based Tactical Strategy Group, Inc, a consulting firm specializing in strategic systems redevelopment planning support and Year 2000 solutions. He is the principal author of TSRM, a product of HCL James Martin Inc.

About TSRM

TSRM Year 2000 enterprise planning scenario includes guidelines for project mobilization, pre-planning setup, inventory, portfolio segmentation, project prioritization and deployment plan development. Implementation scenarios include Year 2000 Date Field Expansion and Year 2000 Procedural Workaround templates. Related scenarios support alternative migration options, stand alone data store upgrades, package assimilation efforts and system consolidation projects.

TESTIMONY OF TERRY J. PIDDINGTON

EXECUTIVE VICE PRESIDENT

CTA INCORPORATED

6116 EXECUTIVE BLVD.

ROCKVILLE, MD 20852

MAY 14, 1996

BEFORE THE

HOUSE SCIENCE COMMITTEE

SUBCOMMITTEE ON TECHNOLOGY

CTA SOLUTIONS FOR THE YEAR 2000 SOFTWARE FOR THE VETERAN'S ADMINISTRATION

AUSTIN, TEXAS

My name is Terry Piddington and I am an Executive Vice President of CTA INCORPORATED, a provider of Information Technology Services to both the Federal and State governments. I am pleased to offer the following testimony regarding our experience with the Year 2000/Century Date Conversion problem that is facing both government and industry.

The problem we are facing today is the direct result of a computer programming convention used during the 60's, 70's, and 80's whereby only two digits were used to represent the year in many computer programs. Computer programs that use this convention for representing dates will not function properly past December 31, 1999. Computer applications ranging from benefits processing and financial forecasting to computing the position of satellites in orbit will be adversely affected. The problem is real and must be dealt with quickly if we are to avoid major disruptions in both government services and commercial commerce as we enter the next millennium.

CTA recently completed a successful Year 2000 conversion project for the Bureau of Veterans Affairs (VA) Austin Automation Center. This project, initiated by the VA in 1993 represents the only such project we are aware of within the Federal government that has been fully and successfully completed. In addition, we are currently engaged in Year 2000 conversion projects for two other Federal agencies. We would like to take this opportunity to share with you the lessons we have learned to date on these projects and to offer several recommendations for further addressing this critical problem.

First and foremost we need to recognize that waiting for a “silver bullet” solution to this problem to materialize could be catastrophic. No such solution exists today and we do not believe that any such solution will be forthcoming. The longer organizations wait before initiating the conversion projects the higher the risk that they will not be completed in time.

Based on our experience, the real challenge is not in the technical complexity of the work that must be accomplished but rather the sheer volume of software that must be analyzed and modified. This challenge is compounded by the fact that much of the software that must be modified is written in the COBOL language and is executed on mainframe computers. As a result of the widespread implementation of client server architectures within both government and industry, the number of computer programmers fluent in mainframe COBOL has significantly diminished. This means that we must find ways to maximize the productivity of the computer programmers that are engaged in these conversion projects.

We must recognize that most large data processing centers are operating at or near capacity and that the Year 2000 conversion projects must be accomplished in a manner that does not adversely impact the ability of these centers to continue performing their required functions. A related issue involves assuring that Year 2000 software modifications interface properly with other modifications to the same software that are often being made in parallel by other programmer teams. This is another way of saying that maintaining positive control of the software configurations during the conversion process is essential.

CTA is addressing these challenges today. We recognize that trying to bring the programmers to the software is not the answer. As I've already stated skilled programmers will be in short supply and in many cases will not be located where we need them. We are addressing this problem by establishing Year 2000 conversion centers. Each center will consist of mainframe computers and teams of programmers experienced and skilled in the Year 2000 code conversion process. In addition, the programmers in each center will be provided with the best automated support tools available to enhance their productivity. CTA has selected the toolset provided by VIASOFT, Inc. for use in our centers. We believe that the use of these tools will increase the productivity of our programmer teams by 20% to 30%. Our first such center will be located in Austin, Texas. This approach will allow us to accomplish the bulk of the necessary conversion work outside of our customers data processing facilities. This approach minimizes interference with our clients' ongoing operations and maximizes the productivity of our skilled programmer teams who are operating in a highly efficient conversion factory environment.

While we in the information technology industry can do much to contribute to the solution to this problem, there are several actions that we believe the Government must take. A factor that serves to compound the Year 2000 conversion problem is the lack of an accepted government standard for representing date fields in computer programs. We encourage the government to step up its efforts to develop and publish such standards while there is still time to implement them. The Government needs to quickly address the problem of how to more rapidly acquire Year 2000 conversion services from industry. There is not enough time available to acquire these services through the standard Federal acquisition process. Finally, we strongly encourage all Federal agencies to begin to aggressively pursue planning and implementation of conversion projects. There's no time to waste.

The problem we are facing is a global problem and we must be realistic about what can be done to solve it. There is not enough time, money, or software programmers on the planet to convert all of the software affected by this problem in the next 1220 days. Therefore, we need to carefully prioritize the work to be done to ensure that critical applications are completed in time and adopt methods for doing the conversions that maximize the amount that can be accomplished in the time available.

I want to thank you for the opportunity to present this testimony

BOSTON PUBLIC LIBRARY
3 9999 05984 249 0

ISBN 0-16-052913-1
90000
9 780160 529139